SCRAMMIKUB

Autor: Pau Sabé Martínez
Director: Enric Mayol Sarroca

Enginyeria de Software
1r quadrimestre – curs 2016/2017
Facultat d'Informàtica de Barcelona (FIB)
Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

19 de gener de 2017

Resum

Cada vegada més, els dispositius mòbils estan més integrats a les nostres vides. Temps enrere era impensable la idea de jugar a jocs a través d'un dispositiu mòbil i menys encara de fer-ho de forma online o contra una màquina intel·ligent.

Aquest projecte consisteix en el desenvolupament, partint d'una planificació acurada prèvia, d'un joc de taula anomenat Scrammikub que permet unir jugadors a través de la xarxa i l'opció de jugar contra el mateix sistema. És doncs un projecte de creació de software de qualitat de principi a fi.

Resumen

Cada vez más, los dispositivos móviles están más integrados en nuestras vidas. Tiempo atrás era impensable la idea de jugar a juegos a través de un dispositivo móvil i menos aun de hacerlo de forma online o contra una máquina inteligente.

Este proyecto consiste en el desarrollo, partiendo de una cuidadosa planificación previa, de un juego de mesa llamado Scrammikub que permite unir jugadores a través de la red i la opción de jugar contra el mismo sistema. Es pues un proyecto de creación de software de cualidad de principio a fin.

Abstract

Increasingly, smartphones are more integrated in our lives. Time ago it was unthinkable the idea of playing games though mobile devices and out of our minds play with people who are connected or even play with virtual machines with artificial intelligence.

This project consists in develop a board game called Scrammikub with a good previous planning. This game allows join players though the network and also the option of play against the own system. Therefore, this is a quality software building project from beginning to end.

ÍNDEX

Índex d'il·lustracions 8

Índex de taules 9

1. Introducció i contextualització 10

1.1. Contextualització 10

1.2. Actors implicats 11

1.2.1. Desenvolupador, dissenyador i beta tester 11

1.2.2. Director del projecte 12

1.2.3. Desenvolpuadors de software 12

1.2.4. Usuaris 12

1.3. Estat de l'art 12

1.3.1. Literatura 12

1.3.2. Estudi de mercat 12

1.4. Formulació del problema 13

1.4.1. Competències tècniques 13

2. Abast del projecte 15

2.1. Abast 15

2.2. Possibles obstacles 16

2.2.1. Errors de codi 16

2.2.2. Limitació temporal 16

2.2.3. Falta de coneixement 17

2.3. Metodologia i rigor 17

2.3.1. Mètode de Treball 17

2.3.2. Eines de seguiment 17

2.3.3. Mètode de validació 17

3. Planificació temporal 18

3.1. Durada estimada del projecte 18

3.2. Descripció de les tasques 18

3.2.1. Cursar GEP 18

3.2.2. Cerca d'informació 18

3.2.3. Configurar entorn de treball 18

3.2.4. Desenvolupament del joc 19

3.2.5. Prova de funcionalitats 19

3.2.6. Redacció de la memòria 20

3.3. Recursos 20

3.3.1. Recursos humans 20

3.3.2. Recursos materials 20

3.4. Seqüència lògica de les tasques 20

3.4.1. Diagrama de GANTT 15

3.5. Valoració d'alternatives i pla d'acció 23

4. Gestió econòmica 24

4.1. Identificació i estimació dels costos 24

4.1.1. Recursos humans 24
4.1.2. Costos directes 25
4.1.3. Costos indirectes 26
4.1.4. Imprevistos 26
4.1.5. Contingència i cost total 26
4.2. Control de gestió 27
5. Sostenibilitat 28
5.1. Dimensió econòmica 28
5.2. Dimensió social 28
5.3. Dimensió ambiental 28
5.4. Matriu de sostenibilitat 29
6. Normativa del joc 30
7. Disseny 32
7.1. Entorn de treball 32
7.1.1. Java 32
7.1.2. Framework LibGDX 32
7.1.3. Google Play Games Services 33
7.1.4. TExture Packer 34
7.1.5. Photoshop 35
7.2. Disseny de l'arquitectura 35
7.2.1. Disseny del servidor 35
7.2.2. Disseny del client 36
7.3. Classes 38
7.3.1. AndroidLauncher 40
7.3.2. mainGame 40
7.3.3. MenuScreen 40
7.3.4. SettingsScreen 41
7.3.5. PlayScreen 41
7.3.6. ControllersFactory 41
7.3.7. PlayController 42
7.3.8. OfflineController 44
7.3.9. Player 46
7.3.10. Board 46
7.3.11. Bag 46
7.3.12. Hand 46
7.3.13. Ball 46
8. Implementació 48
8.1. Tasques d'Scrammikub 48
8.1.1. Fer LogIn / LogOut 48
8.1.2. Sortir de l'aplicació 48
8.1.3. Configurar paràmetres 48
8.1.4. Invitar a jugadors online i veure invitacions 48
8.1.5. Tots els jugadors compartim bossa 49

8.1.6. Posicionar boles al taulell 49
8.1.7. Moure les boles del taulell 49
8.1.8. Desfer una bola 49
8.1.9. Desfer el moviment complet 49
8.1.10. Passar torn 50
8.1.11. Moviment controlat, és correcte o no 50
8.1.12. Agafar una bola de la bossa compartida 50
8.1.13. Jugar amb comodins 50
8.1.14. Sortir del joc 50
8.1.15. Controlat pel temps 50
8.1.16. Ser avisat del que passa pel sistema 51
8.1.17. Veure l'estat de les rondes 52
8.1.18. Veure l'estat dels jugadors, activitat i torn 52
8.1.19. Veure l'estat del temps 52
8.1.20. Veure la finalització de la partida 53
8.1.21. Sistema tancat, sense errors 53
8.1.22. Estètica 53
8.1.23. Jugar en mode online i offline 53

9. Identificació de lleis i regulacions 54
9.1. Tractament de les dades dels usuaris 54
9.2. Software lliure 54

10. Resultats 56
10.1. Treballs futurs 59

11. Conclusions 60

12. Referències 62

13. Annex 64
13.1. Com instal·lar Scrammikub 64
13.2. Com implementar versions futures 64

ÍNDEX D'IL·LUSTRACIONS

Il·lustració 1. Taulell d'Scrabble durant una partida [2] 10

Il·lustració 2. Taulell de Rummikub durant una partida [3] 11

Il·lustració 3. Taulell d'Scammikub durant una partida. 11

Il·lustració 4. Taulell d'Scrammikub durant una partida i inventari de les peces del joc 15

Il·lustració 5. Diagrama de GANTT del projecte 15

Il·lustració 6. Elements d'Scrammikub 31

Il·lustració 7. Consola de desenvolupadors de Google 33

Il·lustració 7. Texture packer de botó Close 34

Il·lustració 8. Esquema de l'arquitectura del servidor 35

Il·lustració 9. Patró MVC [10] 37

Il·lustració 10. Captura del monitor de memòria d'Android Studio 38

Il·lustració 12. Diagrama de classes UML 39

Il·lustració 13. Demo de bucle 42

Il·lustració 14. Flux d'una partida amb jugadors online i màquines 43

Il·lustració 15. Textura de l'estat dels jugadors 52

Il·lustració 16. Textura de l'estat del temps 52

Il·lustració 17. Textura de ronda acabada 53

Il·lustració 18. Resultat: pantalla Menu 56

Il·lustració 19. Resultat: veure invitacions 56

Il·lustració 20. Resultat: veure invitacions 57

Il·lustració 21. Resultat: pantalla Settings 57

Il·lustració 22. Resultat: pantalla Play 57

Il·lustració 23. Resultat: moviment incorrecte 58

Il·lustració 24. Resultat: partida online 58

ÍNDEX DE TAULES

Taula 1. Recursos materials i utilitats 20

Taula 2. Costos dels recursos humans estimats 24

Taula 3. Costos dels recursos humans estimats per tasca 25

Taula 4. Costos directes de software estimats 26

Taula 5. Costos indirectes estimats 26

Taula 6. Cost total del projecte 27

Taula 7. Rang de sostenibilitat 29

Taula 8. Accions de l'usuari a la vista del menú inicial 40

Taula 9. Accions de l'usuari a la vista d'ajustaments 41

Taula 10. Accions de l'usuari a la vista de partida 41

Taula 11. Missatges del sistema 51

1. INTRODUCCIÓ I CONTEXTUALITZACIÓ

Aquest Treball de Final de Grau és de l'especialitat d'Enginyeria del Software de la Facultat d'Informàtica de Barcelona de la Universitat Politècnica de Catalunya. És un projecte de la modalitat A que consisteix en un software que permet a usuaris jugar a un joc anomenat Scrammikub basat en la barreja de dos famosos jocs anomenats *Scrabble* i *Rummikub*.

1.1. CONTEXTUALITZACIÓ

Avui en dia, un 35% [1] de la població utilitza els telèfons mòbils per comunicar-se, estar al dia de les novetats que succeeixen al món, formar part de comunitats cibernètiques i també per entretenir-se jugant a jocs.

En el món dels mòbils hi podem trobar moltes classes de jocs: de taula, de rol, videojocs, etc. Aquest projecte pretén desenvolupar Scrammikub, un joc de taula que té la finalitat de proporcionar, als usuaris que hi juguin, entreteniment i competitivitat.

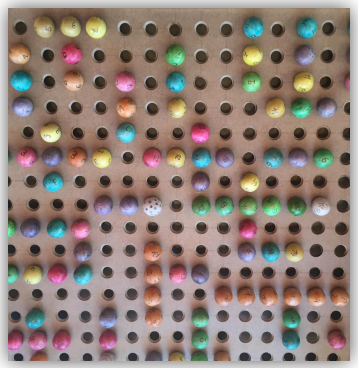
Scrammikub és una combinació de dos jocs. El primer és l'Scrabble que és un joc de taula molt famós que consisteix en que cada jugador aconsegueixi el màxim de puntuació possible. Els jugadors poden obtenir punts en la construcció de paraules dins d'un tauler i unes lletres determinades.

Il·lustració 1. Taulell d'Scrabble durant una partida [2]

El segon és el Rummikub que també és un joc de taula en el qual els jugadors han d'aconseguir obtenir la mínima puntuació possible. En aquest però els usuaris obtindran punts si en acabar la partida no han aconseguit guanyar i tenen fitxes acumulades.

Il·lustració 2. Taulell de Rummikub durant una partida [3]

La mecànica d'Scammikub segueix la de Rummikub però amb la possibilitat del creuament de sèries numèriques tal i com es fa amb les paraules d'Scrabble.



Il·lustració 3. Taulell d'Scammikub durant una partida.

El taulell és doncs cíclic, es poden fer sèries de números sense límit de taulell, el nombre de jugadors és entre 2 i 6 amb edat mínima de 8 anys i les partides tenen una durada de 15 a 20 minuts.

1.2. ACTORS IMPLICATS

Els actors implicats són aquells que es beneficien o estan interessats en la implementació del joc.

1.2.1. DESENVOLUPADOR, DISSENYADOR I BETA TESTER

Totes aquelles tasques necessàries per desenvolupar, dissenyar i provar el software seran realitzades per mi mateix, autor d'aquest document.

1.2.2. DIRECTOR DEL PROJECTE

El director d'aquest projecte és l'Enric Mayol Sarroca, que és l'encarregat de guiar, dirigir i supervisar el desenvolupament del projecte tenint en compte els objectius i els terminis establerts.

1.2.3. DESENVOLPUADORS DE SOFTWARE

Les empreses creadores de jocs podrien estar interessades en el joc per l'amenaça financera que pot comportar l'entrada en mercat del joc i/o l'interès per obtenir els drets del mateix per a poder-lo comercialitzar.

De la mateixa manera, empreses que gestionen sistemes operatius podrien tenir un interès per a introduir el joc en els seus sistemes.

1.2.4. USUARIS

Tota aquella persona que jugui amb aquest joc.

1.3. ESTAT DE L'ART

1.3.1. LITERATURA

La creació d'un videojoc per a dispositius mòbils és quelcom que s'ha fet des de la creació dels primers mòbils. La idea de Scarmmikub però és totalment nova. Tan Rummikub com Scrabble són dos jocs de taula que s'han adaptat a dispositius mòbils de diferents maneres però no és així amb la combinació d'ambdós.

Per tant, la solució d'aquest projecte partirà de zero amb el disseny d'un projecte totalment nou i no serà adaptació d'un altre existent.

1.3.2. ESTUDI DE MERCAT

Les dues grans empreses de sofware de mòbil Apple amb iOs i Google amb Android disposen d'aplicacions similars a les que aquest projecte pretén desenvolupar.

Les aplicacions per a mòbils, com també els jocs, estan altament presents a la societat a qui va dirigida Scrammikub. Al 2014 hi havia un total de 1,21 milions a Apple Store [4] de Alppe i 1,43 miliions d'aplicacions al Play Store [5] de Google. [6]

Entre Apple Store i Google Play hi podem trobar més de 40 aplicacions considerades similars, ja que és pot jugar al Rummikub o a l'Scrabble de forma multijugador online o offline. Una d'aquestes n'és *Rummikub* [7] que permet jugar a Rummikub amb mode multijugador online o offline en dispositius Android.

1.4. FORMULACIÓ DEL PROBLEMA

El projecte té com a objectiu crear un software capaç de proporcionar a usuaris de dispositius mòbils la possibilitat de jugar al joc Scrammikub.

Podem dividir aquest objectiu en diferents parts:

1- Identificar quin perfil d'usuari li pot interessar aquest software.
2- Donat el perfil d'usuari el qual volem satisfer pensar el disseny del software que es necessita.
3- Desenvolupar el software dissenyat.

El desenvolupament del software té la complexitat de resoldre el problema que comporta la multiconexió d'usuaris online, la creació d'un mòdul d'intel·ligència artificial que permeti el mode offline, i també aconseguir bona jugabilitat, tenint en copte que el taulell d'Scrammikub no és pas petit.

1.4.1. COMPETÈNCIES TÈCNIQUES

A continuació es mostren les competències tècniques que s'ha escollit per aquest projecte i el seu nivell d'assoliment.

CES1.1: *Desenvolupar, mantenir i avaluar sistemes i serveis software complexos i/o crítics.*

S'espera assolir un nivell alt d'aquesta competència ja que Scrammikub és un software complex i aquest projecte consisteix en desenvolupar-lo.

CES1.2: *Donar solució a problemes d'integració en funció de les estratègies, dels estàndards i de les tecnologies disponibles.*

S'espera assolir un nivell baix d'aquesta competència ja que no es creu que hi hagi molts problemes d'integració. Malgrat això, se sap que n'hi ha (disseny d'arquitectura, del servidor, etc) i per això s'accepta com a competència.

CES1.4: *Desenvolupar, mantenir i avaluar serveis i aplicacions distribuïdes amb suport de xarxa*.

S'espera assolir un nivell alt d'aquesta competència ja que Scrammikub és un software amb suport online i aquest projecte pretén desenvolupar-lo.

CES1.7: *Controlar la qualitat i dissenyar proves en la producció de software.*

S'espera assolir un nivell normal d'aquesta competència ja que malgrat es prioritzi més que el sistema funcioni, no es deixa de banda que ha de ser de qualitat.

CES1.8: *Desenvolupar, mantenir i avaluar sistemes de control i de temps real.*

S'espera assolir un nivell baix d'aquesta competència ja que una part del sistema és de control i de temps real.

CES2.1: *Definir i gestionar els requisits d'un sistema software.*

S'espera assolir un nivell baix d'aquesta competència ja que malgrat es prioritzi més que el sistema funcioni se sap que és necessària la definició i gestió dels requisits del mateix.

CES3.1: *Desenvolupar serveis i aplicacions multimèdia.*

S'espera assolir un nivell alt d'aquesta competència ja que Scrammikub és una aplicació multimèdia i aquest projecte pretén desenvolupar-lo.

2. ABAST DEL PROJECTE

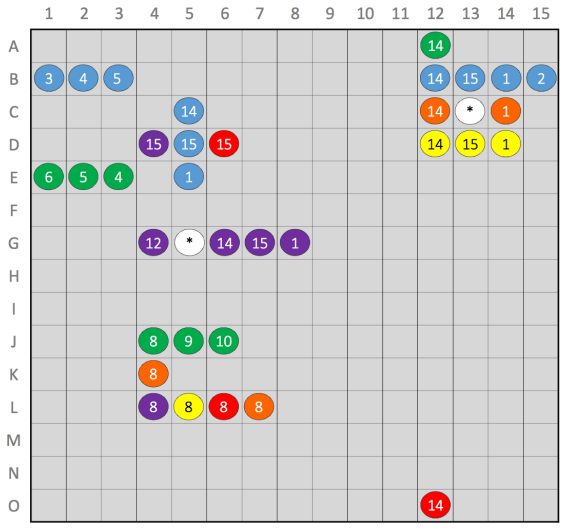
2.1. ABAST

Per a la bona realització del projecte cal tenir clar el funcionament del joc. Així doncs, seguidament estarà descrita la mecànica del joc així com les seves implementacions i funcionalitats inicials i bàsiques.

Els jugadors comencen la partida amb 15 boles amb números a l'atzar. Un jugador podrà fer la primera jugada (posar boles al taulell) quan ho faci amb una combinació que li proporcioni 15 punts. Només tindrà un minut per elaborar cada jugada i en cas que en una jugada no pugui posar cap bola al taulell podrà agafar-ne una altre a l'atzar.

Les combinacions de boles són de 3 boles mínim les quals poden ser escales del mateix color o boles amb el mateix número i color estrictament diferents. El taulell és cíclic i per tant les combinacions poden ser dipositades al taulell de manera que el final del taulell d'un costat és l'inici del costat oposat. A més existeixen comodins que permeten simular el número i color desitjat.

El sistema de puntuació funciona de tal manera que quan s'acaba la partida el guanyador suma 0 mentre que els altres jugadors sumen tants punts com és la suma de les seves boles no baixades al taulell (els comodins sumen 50).



INVENTARI	
15 30 boles blaves en dues sèries numerades de l'1 al 15.	15 30 boles verdes en dues sèries numerades de l'1 al 15.
15 30 boles vermelles en dues sèries numerades de l'1 al 15.	15 30 boles taronges en dues sèries numerades de l'1 al 15.
15 30 boles grogues en dues sèries numerades de l'1 al 15.	* 6 boles blanques decorades que seran els comodins.
15 30 boles liles en dues sèries numerades de l'1 al 15.	Rellotge de sorra de 1, 2 i 3 minuts.

Il·lustració 4. Taulell d'Scrammikub durant una partida i inventari de les peces del joc

Hi haurà dos tipus de jugabilitats. La primera és aquella la qual només serà d'una sola partida i la segona que et permetrà jugar-ne varies amb l'opció d'anar acumulant punts i així determinar el guanyador en funció del qui té menys punts en un nombre determinat de partides seguides.

Amb el funcionament del joc entès ja es podrà començar amb els subobjectius descrits en la *formulació del problema*. Primer identificant el tipus d'usuari al qual interessa satisfer amb aquest software i després la implementació d'aquest responent a les seves necessitats.

El projecte doncs serà principalment la creació d'una aplicació que serà per a Android desenvolupada amb l'entorn de desenvolupament Android Studio, Java com a llenguatge principal de programació que permetrà a l'usuari jugar a aquest joc de forma online i offline.

2.2. POSSIBLES OBSTACLES

Projectes d'aquesta magnitud sovint tenen obstacles en el seu procés de desenvolupament. Seguidament se n'anomena els que principalment poden afectar i les seves solucions.

2.2.1. ERRORS DE CODI

Els errors de codi són aquells errors identificats en el codi que no permeten un bon funcionament del software. Aquests són molt freqüents en els projectes de creació de software.

Per evitar-los es farà una estricta avaluació minuciosa cada cop que s'hagi implementat una funcionalitat del projecte.

2.2.2. LIMITACIÓ TEMPORAL

Només es disposa de 3 a 4 mesos per a la realització del projecte.

Per evitar aquest obstacle caldrà realitzar un estricte calendari que permeti la distribució equitativa de tot el pes del treball en tot aquest període de temps.

2.2.3. FALTA DE CONEIXEMENT

El desenvolupador d'aquest projecte desconeix com funcionen els sistemes multijugadors així com la tecnologia emprada per a la comunicació de bases de dades. Caldrà doncs un previ estudi d'aquests coneixements.

2.3. METODOLOGIA I RIGOR

2.3.1. MÈTODE DE TREBALL

Per a la realització d'aquest projecte s'utilitzarà la metodologia de treball SCRUM que té com a estratègia dividir el pes del projecte en tasques les quals seran desenvolupades i integrades en moments diferents.

2.3.2. EINES DE SEGUIMENT

Per poder guardar el codi per evitar pèrdues no desitjades de codi o tornar en estats de codi anteriors s'utilitzarà el control de versions Git [8].

Per altra banda, també hi haurà reunions amb el director del treball per tal de tenir un seguiment i *feedback* del treball. Igualment serà freqüent l'ús del correu electrònic per facilitar això mateix.

2.3.3. MÈTODE DE VALIDACIÓ

El director servirà com a mètode de validació del projecte ja que validarà, reconduirà i guiarà durant tot el procés del treball.

Per la validació del codi del projecte s'avaluarà el funcionament de cada funcionalitat descrita cada vegada que se n'afegeixi una de nova. Aquest procés es realitzarà manualment resseguint cada una de les funcionalitats i comprovant el seu correcte funcionament.

3. PLANIFICACIÓ TEMPORAL

3.1. DURADA ESTIMADA DEL PROJECTE

El projecte té una durada estimada d'uns 4 mesos, des de mitjans de setembre de 2016 a mitjans de gener de 2017.

3.2. DESCRIPCIÓ DE LES TASQUES

Seguidament es descriu la divisió de tasques del projecte:

3.2.1. CURSAR GEP

Aquesta tasca consisteix en cursar Gestió de Projectes (GEP) on és desenvolupa la major part de l'estudi preliminar del treball. S'hi estudia conceptes tals com l'abast del projecte, contextualització, planificacions, gestió econòmica, etc.

No hi ha dependència de precedència amb altres tasques.

3.2.2. CERCA D'INFORMACIÓ

Aquesta tasca consisteix en determinar, a partir de la cerca d'informació, quin és el millor camí per aconseguir l'objectiu del treball. Per tant, determinar quin és el millor entorn de treball per desenvolupar el joc.

No hi ha dependència de precedència amb altres tasques.

3.2.3. CONFIGURAR ENTORN DE TREBALL

En aquesta tasca es posarà apunt l'entorn de treball més adequat. L'entorn de treball consisteix en totes aquelles eines necessàries, com podria ser l'ordinador i el software per programar el joc.

Té com a dependència de precedència la cerca d'informació per saber quin és l'entorn de treball òptim.

3.2.4. DESENVOLUPAMENT DEL JOC

Tot seguit es mostren les subtasques previstes. Estan amb ordre de prioritat però això no vol dir que es desenvolupin estrictament de manera seqüencial. Cada subtasca és una funcionalitat del joc, això és clau en la gestió del temps tenint en compte que la metodologia de treball SCRUM que s'utilitza.

Té com a dependència de precedència la configuració de l'entorn de treball necessari per a poder desenvolupar-ho.

3.2.4.1. POSADA APUNT DE LA CONEXIÓ AMB LA BASE DE DADES

Creació de la base de dades amb els adaptadors necessaris per a la seva connexió des del codi.

3.2.4.2. GESTIÓ D'USUARIS

Posada apunt de la gestió d'usuaris i la seva corresponent interacció amb la base de dades.

3.2.4.3. INTERFÍCIE GRÀFICA MÍNIMA

Interfície gràfica del joc. Durant tot el desenvolupament s'hi anirà introduint canvis, segons les funcionalitats afegides.

3.2.4.4. MECÀNICA BÀSICA DEL JOC

Desenvolupament del joc amb les seves característiques (taulell, fitxes, interacció d'usuaris, etc).

3.2.4.5. MILLORA D'INTERFÍCIE GRÀFICA I MECÀNICA DEL JOC

Tasca encarregada de desenvolupar una interfície gràfica més agradable i usable per l'usuari i una mecànica de joc més elaborada amb certes millores de jugabilitat.

3.2.5. PROVA DE FUNCIONALITATS

Prova de cada funcionalitat afegida i desenvolupada per comprovar el correcte funcionament de les afegides anteriorment i les noves.

És necessari haver acabat cada funcionalitat per provar-la, per tant té com a dependència de precedència la funcionalitat a provar.

3.2.6. REDACCIÓ DE LA MEMÒRIA

Supervisió de la feina feta en la tasca 3.2.1 (Cursar GEP) i afegir la descripció del procés del projecte.

És necessari haver acabat la resta de tasques per començar aquesta, així que hi ha una dependència de precedència amb la resta de tasques.

3.3. RECURSOS

3.3.1. RECURSOS HUMANS

El projecte serà dut a terme per una sola persona encarregada de plantejar, analitzar, dissenyar, desenvolupar i provar el joc.

3.3.2. RECURSOS MATERIALS

Per dur a terme les tasques es disposarà dels següents recursos materials:

Recurs	Utilitat
MacBook Pro. *Intel Core i5 de doble núcleo a 2,7 GHz, 8 GB de memoria LPDDR3 a 1.866 MHz i MacOS Sierra*	Desenvolupament del joc i la memòria
Microsoft Office Word per Mac 15.22	Redacció de la memòria
Android Studio 2.2	Entorn de desenvolupament
Gmail	Comunicació
BitBucket	Control de versions del codi

Taula 1. Recursos materials i utilitats

3.4. SEQÜÈNCIA LÒGICA DE LES TASQUES

Les metodologies àgils com SCRUM, estratègia seleccionada per a la realització del projecte, no pretenen tenir una planificació exacte i completa del procés previ. SCRUM funciona de manera incremental amb la possibilitat de crear canvis durant el projecte si és necessari. Així doncs, no s'intentarà dissenyar una seqüència molt acurada de l'ordre de desenvolupament de les tasques.

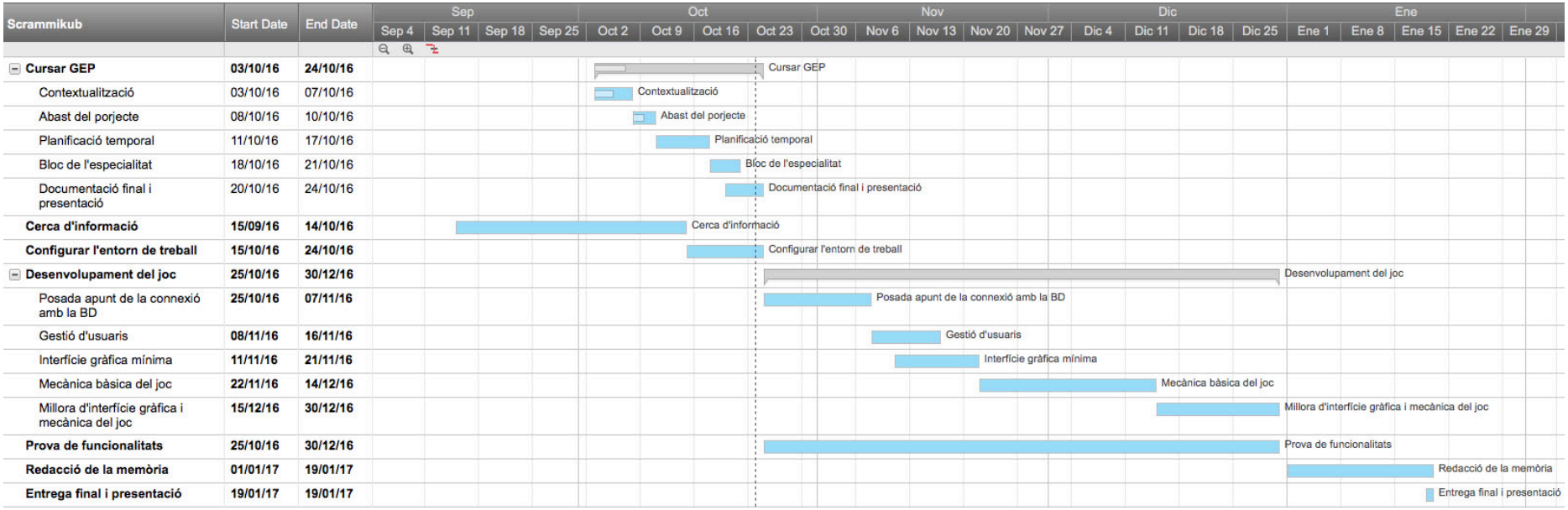
Tanmateix, tenint en compte les dependències de precedència pròpies de les tasques es pot determinar un cert ordre lògic i repartit en el temps disponible. És el següent:

Les tasques 3.2.1 (Cursar GEP) i 3.2.2 (Cerca d'informació) comencen les primeres i de forma paral·lela. Un cop finalitzada la 3.2.2 (Cerca d'informació) es podrà procedir amb la 3.2.3 (Configurar l'entorn de treball). Un cop finalitzada es podrà procedir amb la 3.2.4 (Desenvolupament del joc) conjuntament amb la 3.2.5 (Prova de funcionalitats). I un cop finalitzades es podrà dur a terme l'última, 3.2.6 (Redacció de la memòria).

3.2.1 (Cursar GEP) té inici a mitjans de setembre i fi a finals d'octubre. 3.2.2 (Cerca d'informació), 3.2.3 (Configurar l'entorn de treball), 3.2.4 (Desenvolupament del joc) i 3.2.5 (Prova de funcionalitats) tenen inici a mitjans de setembre i fins dues setmanes abans de l'entrega del projecte, al gener.

3.2.6 (Redacció de la memòria) té inici a principis de gener i fi a mitjans de gener amb l'entrega final del projecte.

3.4.1. DIAGRAMA DE GANTT



Il·lustració 5. Diagrama de GANTT del projecte

En aquest diagrama de GANTT es pot observar el desglossament de les tasques en el temps que s'ha exposat anteriorment.

3.5. VALORACIÓ D'ALTERNATIVES I PLA D'ACCIÓ

En un projecte com aquest és inevitable que hi hagi imprevistos que puguin fer perillar l'acabament del mateix en el temps previst.

Utilitzant una metodologia àgil i tenint en compte la data d'entrega del projecte és necessari tenir clar quines funcionalitats es vol aconseguir mínimes abans de termini i quines serien extres. D'aquesta manera es pot avaluar com és l'estat del projecte amb més precisió mirant quines són les funcionalitats del joc desenvolupades fins al moment i quines falten per desenvolupar.

Aquestes funcionalitats, descrites anteriorment en el punt 3.2.4, tenen una estimació de temps equitativa i possibles de ésser desenvolupades en el termini establert per a un sol treballador, com és el cas. D'aquesta manera s'assegura un control del consum de recursos disponibles.

Les funcionalitats del joc mínimes haurien de ser aquelles que es poden fer de manera còmode amb el temps disposat. Les extres haurien de ser aquelles que teòricament, amb el temps disponible, es podrien implementar però que en cas d'algun contratemps es podrien no fer. En aquest cas n'hi ha una, 3.2.4.5 (Millora d'interfície gràfica i mecànica del joc).

D'aquesta manera es pot determinar que el projecte podrà ser acabat, malgrat possibles imprevistos, en el temps previst.

4. GESTIÓ ECONÒMICA

4.1. IDENTIFICACIÓ I ESTIMACIÓ DELS COSTOS

Tot seguit s'analitzen i s'estimen els recursos necessaris per a la realització del projecte: recursos humans, costos directes i costos indirectes.

4.1.1. RECURSOS HUMANS

Malgrat tot el projecte el projecte el desenvolupa una mateixa persona, aquesta tindrà diferents rols. A continuació es mostren els pressupostos estimats segons aquests rols:

ROL	ÀLIES	€ PER HORA	HORES	COST
CAP DE PROJECTE	C	45€	120	5.400,00€
ANALISTA	A	25€	40	1.000,00€
DISSENYADOR	D	30€	60	1.800,00€
DESENVOLUPADOR	V	23€	260	5.980,00€
TESTEJADOR	T	15€	60	900,00€
TOTAL		-	520	15.080,00€

Taula 2. Costos dels recursos humans estimats

La següent taula es mostra aquest pressupost estimat però tenint en compte les tasques definides anteriorment en el diagrama de GANTT:

NOM DE LA TASCA	HORES ESTIMADES	ROL	COST
GESTIÓ DE PROJECTES	86	C	3.870,00€
CONTEXTUALITZACIÓ	20		
ABAST DEL PROJECTE	12		
PLANIFICACIÓ TEMPORAL	22		
BLOC DE L'ESPECIALITAT	19		
DOCUMENTACIÓ FINAL I PRESENTACIÓ	15		
CERCA D'INFORMACIÓ	20	A	500,00€
CONFIGURAR ENTORN DE TREBALL	18	V	414,00€
DESENVOLUPAMENT DEL JOC	222	V	5.106,00€
POSADA APUNT DE LA CONNEXIÓ AMB LA BD	34		
GESTIÓ D'USUARIS	35		
INTERFÍCIE GRÀFICA MÍNIMA	45		
MECÀNICA BÀSICA DEL JOC	65		

MILLORA D'INTERFÍCIE GRÀFICA I MECÀNICA DEL JOC	43		
PROVA DE FUNCIONALITATS	60	T	900,00€
REDACCIÓ DE LA MEMÒRIA	80	C+A+D+V	4.290,00€
TOTAL			15.080,00€

Taula 3. Costos dels recursos humans estimats per tasca

4.1.2. COSTOS DIRECTES

4.1.2.1. HARDWARE

Pel càlcul del cost dels recursos físics cal tenir en compte factors com l'amortització del recurs i el temps que s'utilitzarà en aquest treball. Entenem com amortització el temps de vida útil estimat del recurs i el temps emprat és de 5 mesos.

Tot seguit es mostra com s'ha calculat aquests cost i el seu resultat.

$$Cost = Preu * Amortització * \frac{5\ mesos}{12\ mesos/any}$$

L'únic recurs hardware necessari és l'ordinador MacBook Pro. El seu preu és de 1.550€ amb una amortització de 5 anys.

Aplicant la fórmula tenim que el cost directe de hardware estimat és de 129,17€.

4.1.2.2. SOFTWARE

En el cas del software l'amortització és més complexa de calcular ja que surten noves versions del mateix o apareixen softwares similars que potser interessen més. Així que suposarem 3 anys d'amortització per cada programa.

RECURS	PREU	AMORTITZACIÓ	COST ESTIMAT
MAC OS SIERRA	-	-	0,00€
ANDROID STUDIO	0,00€	-	0,00€
GMAIL	0,00€	-	0,00€
BITBUCKET	0,00€	-	0,00€
LIBGDX	0,00€	-	0,00€
TEXTURE PACKER	0,00€	-	0,00€

MICROSOFT OFFICE MAC	149,00€	3	20,69€
PHOTOSHOP	24,19€/mes	3	120,95€
TOTAL			141,64€

Taula 4. Costos directes de software estimats

4.1.3. COSTOS INDIRECTES

A més a més, existeixen costos indirectes en el desenvolupament del projecte. Són costos de recursos que actuen indirectament en aquest procés. Un exemple és consum elèctric, que l'estimarem a 2kWh diaris durant els 5 mesos.

RECURS	PREU	QUANTITAT	COST ESTIMAT
LLOGUER OFICINA	430€/mes	5 mesos	2.150,00€
CONSUM ELÈCTRIC	0,15€/mes	300kWh	45,00€
ADSL	44,00€/mes	5 mesos	220,00€
COMPTE ANDROID DEVELOPER	22,45€	-	22,45€
TOTAL			2.437,45€

Taula 5. Costos indirectes estimats

4.1.4. IMPREVISTOS

Bàsicament hi ha dos imprevistos: el retard en el temps de les tasques, el qual es defineix un risc de 35% que això succeeixi, i l'avaria d'un component hardware, amb risc de 5%.

El cost estimat de cada dia de retard serà de 35%*(suma dels costos)/5 mesos, 41,51€. I el cost estimat de l'avaria de 5%*(cost total hardware), 6,46€.

Tenint en compte la previsió d'uns 7 dies de retràs possible, el cost dels imprevistos estimats total és de 41,51€ * 7 + 6,46€, 297,00€.

4.1.5. CONTINGÈNCIA I COST TOTAL

L'estimació dels costos és molta curada, això significa que el valor reservat de contingència pot ser menor, un 8% del cost total.

NOM DEL COST	COST
RECURSOS HUMANS	15.080,00€
DIRECTES	270,81€
INDIRECTES	2.437,45€
IMPREVISTOS	297,00€

COST PARCIAL	18.085,26
CONTINGÈNCIA	8%
COST TOTAL (IVA INC)	**19.532,08€**

Taula 6. Cost total del projecte

4.2. CONTROL DE GESTIÓ

S'ha de preveure que en la realització del projecte es poden produir desviacions. S'ha de tenir una bona planificació per evitar aquests desviaments del pressupost necessari estimat.

Els següents indicadors ajuden a calcular aquestes desviacions:

- Desviació temporal. Final de la tasca prevista menys el final real de la mateixa.
- Desviació del preu d'una tasca. Diferència de preu previst i real de la tasca.
- Desviació del preu d'un recurs. Diferència de preu previst i real del recurs.
- Desviació econòmica total. Diferència de preu previst i real de tot el projecte.

5. SOSTENIBILITAT

5.1. DIMENSIÓ ECONÒMICA

El projecte intenta ser sostenible econòmicament. En el pressupost estimat dels costos sempre s'ha buscat ajustar el preu així que no s'esperen grans despeses econòmiques.

El que incrementa més el cost del projecte és sens dubte els recursos humans i està clar que són clau per a la realització del projecte i no poden ser reduïts.

El temps del projecte podria ser menor si es contractés més treballadors per a realitzar la feina paral·lelament. Això però, segurament encariria el preu.

El projecte no té prevista la col·laboració amb alguna altre entitat acadèmica o d'empresa.

5.2. DIMENSIÓ SOCIAL

Aquest projecte s'està desenvolupant a Catalunya. Aquest país està dins del primer món, baixa pobresa i moltes oportunitats de creixement. Això implica que sigui un lloc favorable a rebre el que el projecte ofereix, ja que consisteix en un joc per dispositius mòbils i per a poder-lo jugar es necessita d'una població que disposi d'aquests dispositius i estigui acostumada a utilitzar-los per a jugar. És obvi que un país del primer món és òptim.

Els consumidors principals del producte a construir són tots aquells que els hi agradin els jocs de taula. A través del producte, aquests consumidors podran guanyar en qualitat de vida ja que podran jugar als jocs de taula però en qualsevol lloc i moment del dia des del dispositiu mòbil.

5.3. DIMENSIÓ AMBIENTAL

Els recursos hardware i software no només s'utilitzaran per aquest projecte. Sent així, s'entén el consum d'aquests recursos com una forma de reutilització d'aquests i se'ls aprofita el més possible, sense gastar-los innecessàriament.

El projecte consisteix en el desenvolupament d'un software i per tant el consum d'electricitat serà el principal efecte ambiental previst.

5.4. MATRIU DE SOSTENIBILITAT

La següent taula mostra, seguint les descripcions de les diferents dimensions, la matriu de sostenibilitat del projecte:

	PROJECTE EN PRODUCCIÓ	VIDA ÚTIL I RESULTATS	RISCS
AMBIENTAL	Anàlisi de recursos	Consum de recursos	Prejudicis ambientals
	10	15	0
ECONÒMIC	Viabilitat econòmica	Cost final	Riscs econòmics
	7	15	0
SOCIAL	Impacte personal	Impacte social	Prejudicis socials
	8	16	0
RANG DE SOSTENIBILITAT	25	46	0
		70	

Taula 7. Rang de sostenibilitat

6. NORMATIVA DEL JOC

Donada la complexitat normativa del joc, s'ha cregut necessari un apartat que expliqui en detall les normes del joc estipulades abans i durant el procés del desenvolupament del joc. D'aquesta manera serà més fàcil entendre alguns dels següents apartats.

Per començar, Scrammikub és un joc de taula on els elements que interactuen són un taulell de 15 per 15 posicions i unes boles amb números i de colors diferents. En el codi es representen els elements d'aquesta manera:

- **Bossa**: lloc amb totes les boles del joc i que disminueix quan un jugador n'agafa alguna.
- **Mà**: boles que un jugador té. La resta de jugadors no les veu.
- **Taulell**: zona per col·locar les boles de 15x15 de tipus cíclic, el final del taulell és el principi del mateix.
- **Bola**: element amb el que el jugador construeix conjunts.
- **Temps**: límit d'un jugador d'efectuar el seu moviment.
- **Rondes**: nombre de partides d'un joc.
- **Jugador**: usuari online o màquina artificial que juga al joc, màxim 6.

Hi ha un total de 6 colors diferents de les boles (blau, vermell, groc, lila, verd i taronja). Per cada color existeixen dues sèries numerades d'1 a 15. A més hi ha 6 comodins que poden ser utilitzats pel color i número que interessi al jugador (malgrat no el pot utilitzar per simular dues boles diferents a la vegada).

Depenent del nombre de jugadors de la partida, la bossa contindrà més o menys boles. Si són de 2 a 4 jugadors, 4 colors de boles (amb les seves sèries numèriques) i 4 comodins. Si són 5 jugadors, 5 colors i 5 comodins. Si són 6 jugadors, 6 colors i 6 comodins.

Com a jugador l'objectiu principal del joc és quedar-se sense boles a la mà. Inicialment es comença amb 15 boles i a cada torn se'n pot agafar una altre.

Per poder desfer-se de les boles de la mà, el jugador pot crear conjunts de boles amb les boles que tingui a la mà i amb les que hi hagi al taulell, però en aquest últim cas el jugador haurà de deixar totes les boles del taulell en algun ordre correcte o altrament no serà vàlid. Si aconsegueix fer un conjunt i hi ha espai en el taulell, el podrà posar. A més, el primer conjunt que baixi ha de ser de valor (suma dels números de les boles) igual o major a 30.

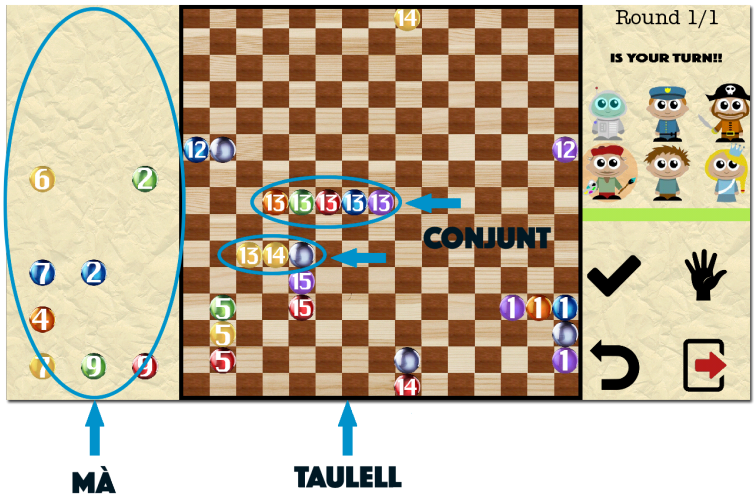
Es té un conjunt quan es forma una sèrie de mínim de 3 boles que compleix que totes elles siguin del mateix color i de número incremental (2, 3, 4 ... o 4, 3, 2 ...) o també si

tenen el mateix número i el color estrictament diferent. Les sèries numèriques són cícliques i quan s'arriba al 15 es pot continuar a l'1 i a l'inrevés.

El jugador haurà d'efectuar el seu moviment dins el temps marcat. Durant aquest temps podrà agafar una bola de la bossa, posar boles de la seva mà al taulell (en forma de conjunt o afegint-les a continuació dels conjunts existents al taulell) i podrà moure boles del taulell per afavorir possibilitats de moviment. També podrà passar el torn al següent usuari però haurà d'haver posat alguna bola de la mà al taulell o haurà d'haver agafat una bola de la bossa, altrament no ho podrà fer. Si se li acaba el temps, se li desfarà automàticament el moviment fet i si no havia agafat bola se li assignarà una.

Es pot configurar certs elements: el nombre de màquines offline (per defecte 3), el nombre de màquines online (per defecte 0), el temps (per defecte 1 minut) i el número de rondes per partida (per defecte 1). Si aquesta última configuració es canvia, cada partida serà una ronda i no acabarà el joc fins que es completin totes. El jugador amb el major nombre de victòries guanyarà el joc[1].

El joc pot jugar-se en mode online contra jugadors i màquines o en offline, només màquines. Cronometrant el joc, té una duració d'entre 10 i 20 minuts cada partida però depèn molt del mode de joc, les partides online solen durar més. A continuació és mostren algunes captures del joc on es veuen els elements descrits:



Il·lustració 6. Elements d'Scrammikub

[1] Aquesta norma, tal i com s'explica al punt de *Treballs futurs*, s'intentaria canviar a que s'acumuli el valor numèric a cada ronda i que quan s'acabessin, guanyés el de menor número.

7. DISSENY

7.1. ENTORN DE TREBALL

Tal i com s'especifica en l'apartat de *Recursos* es necessita de software específic per poder dur a terme el desenvolupament del software. Tot seguit s'explica la raó de la seva elecció i una petita descripció d'aquelles que han tingut un major pes durant el procés de desenvolupament del joc.

7.1.1. JAVA

Igual que la majoria de programes d'Android, Scrammkub també s'ha desenvolupat amb el llenguatge de programació Java. Aquest és de programació orientada a objectes i facilita el tipus de treballs com el d'aquest projecte. La programació orientada a objectes ofereix la possibilitat de dissenyar software de manera que diferents tipus de dades estiguin unides en les seves operacions. Les dades o informació i el codi quedes unides en el que es diuen objectes.

En aquest projecte s'ha necessitat un llenguatge que faciliti la gestió i el maneig del codi, amb qualitat i sense tenir molts problemes. A més Java permet la reutilització de codi mitjançant aquests objectes i tenint en compte l'envergadura del projecte és quelcom no menyspreable.

7.1.2. FRAMEWORK LIBGDX

LibDGX és un framework lliure multiplataforma que funciona en Java. Disposa de funcionalitats de cada plataforma (Android, iOs, Linux, Mac i Windows) per desenvolupar-hi aplicacions. Amb una eina com aquesta s'evita gastar molt de temps per crear aplicacions natives a cada plataforma. Malgrat l'abast del projecte no plantegi més plataformes que Android, és bo que el projecte doni la possibilitat de ser ampliat en un futur i quina millor manera que fer disponible el joc a més plataformes.

Apart de la possibilitat multiplataforma que ofereix, LibGDX no deixa de ser una excepcional eina per crear videojocs en 2D. Les llibreries que ofereix es poden utilitzar per detectar col·lisions, esdeveniments, renderització, etc. LibGDX funciona de tal manera que si vols utilitzar-la pots però no n'estàs obligat. Sobre el codi, això és tradueix en que disposem d'un espai principal anomenat Core on es té tot el material comú a totes les plataformes i un espai individual de cada plataforma on s'hi guarda les classes i operacions específiques a cada una. En aquest projecte es treballa sobre Core i sobre l'espai Android i si es volgués ampliar a iOs, per exemple, s'hauria de traduir la part Android i ja funcionaria.

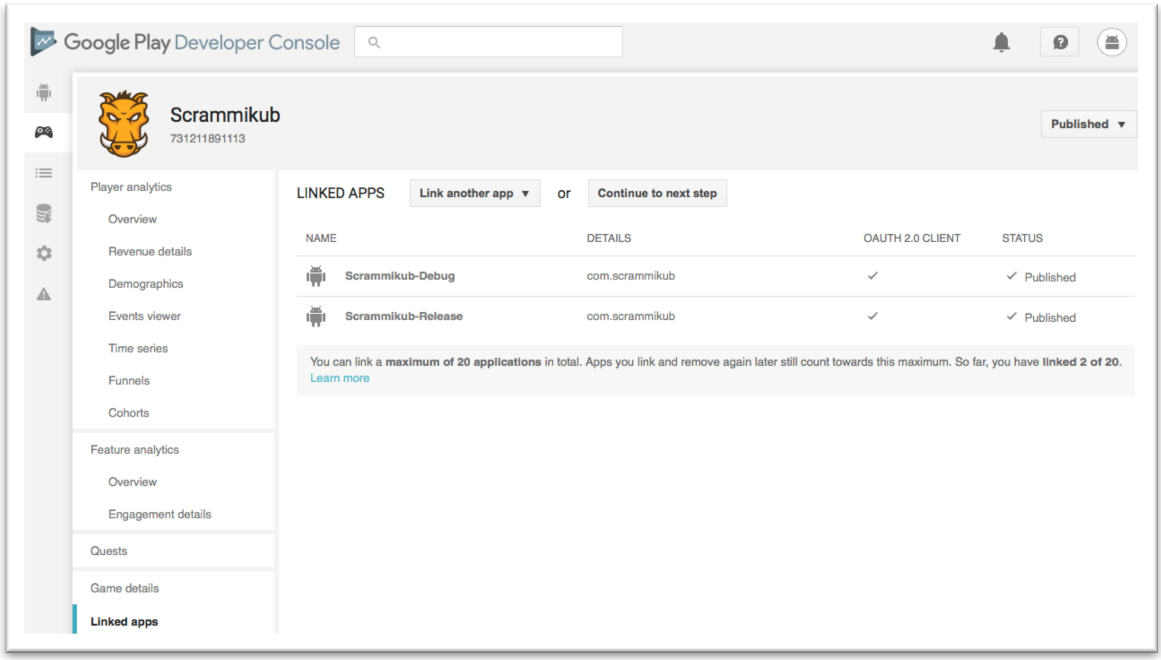
Donades aquestes dues qualitats, es veu aquest framework com a una bona eina per al desenvolupament del software.

7.1.3. GOOGLE PLAY GAMES SERVICES

Google ofereix Google Play Games[9] com a servei de jocs online i API[2] per dispositius mòbil. A través d'aquest servei, el qual Google n'és propietari, es té a l'abast i amb facilitat d'ús totes les eines necessàries per a crear un joc online. En general, la majoria de jocs online comparteixen una sèrie de característiques tals com que hi hagi una sala o habitació virtual en la que hi siguin els jugadors i aquests puguin enviar-se missatges entre ells.

Aquesta ofereix concretament dos usos als seus desenvolupadors. Si s'instal·la l'API que t'ofereix Google es pot elegir configurar-la per ser un joc a temps real o un de torn per torn asíncron. Pel cas d'Scrammikub es necessita el mode a temps real ja que per començar, el joc va per temps i necessita ser síncron.

Per integrar aquesta API es necessita crear un compte de desenvolupador de Google. Així, amb aquest compte, pots accedir a una consola de control de les aplicacions creades amb un apartat destinat als jocs online. La següent imatge mostra aquesta secció de la consola:



Il·lustració 7. Consola de desenvolupadors de Google

[2] Application Programming Interface

A més de configurar aquesta consola, és necessari instal·lar paquets específics proporcionats per Google pel teu IDE (en aquest cas Android Studio) i instal·lar unes llibreries anomenades BaseGameUtils també proporcionades per Google.

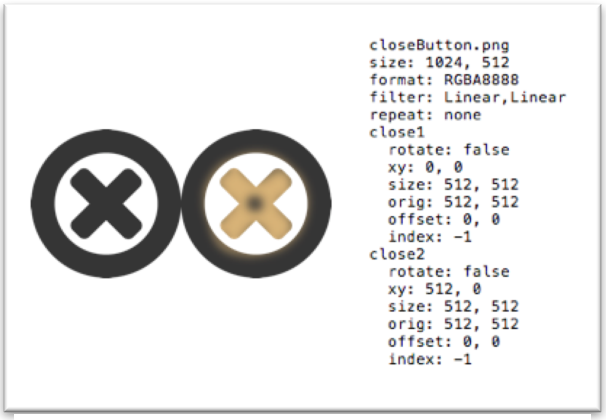
Amb això i la implementació corresponent de l'API, el software ja és capaç de fer logIn, logOut, convidar jugadors, veure invitacions, configurar una habitació virtual, connectar o desconnectar-t'hi i enviar missatges entre jugadors de la mateixa habitació.

L'opció d'utilitzar aquesta eina per al projecte és adequada ja que permet facilitar la implementació i l'ús de l'arquitectura del servidor. De fet, ho facilita fins el punt de no necessitar tractar cap mena d'informació, no cal gestionar cap base de dades ni preocupar-se per les seves crides i té recursos (bona velocitat, límit de transmissió d'informació, etc) de sobres per poder implementar el joc o futures ampliacions.

7.1.4. TEXTURE PACKER

Cada botó del joc ha necessitat de la construcció d'un paquet que contenia les textures que es veien en ser premuts. Texture Packer és un software que construeix una textura a partir d'altres i guarda les coordenades de cadascuna. També guarda quan han de ser mostrades perquè, quan es demani l'objecte que manegi el paquet, sàpiga amb rapidesa i poc cost què ha de mostrar. A més, li facilita molt al dissenyador gràfic de del projecte, ja que aquesta aplicació permet veure com queda l'acció del botó de forma visual i a l'instant.

Scrammikub té molts botons per accions diferents, Texture Packer és doncs una eina imprescindible, especialment per la part gràfica del joc. A continuació es mostra un exemple d'aquest paquet, la seva part gràfica i la seva part de codi:



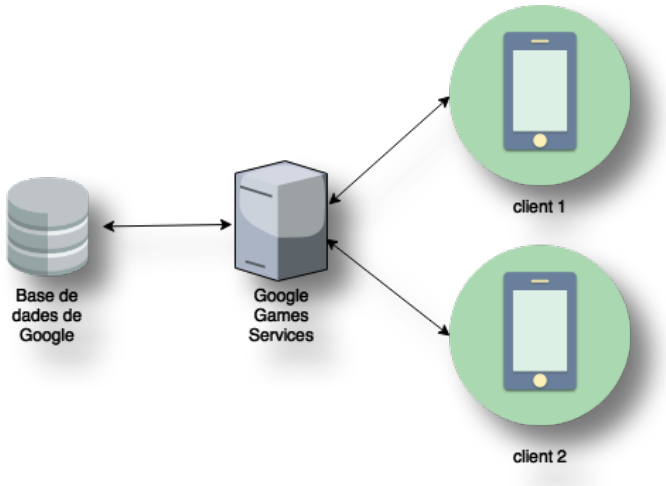
Il·lustració 8. Texture packer de botó Close

7.1.5. PHOTOSHOP

Sense entrar en detalls, Photoshop permet la creació i edició d'imatges. Sense tenir-se molt en compte al principi, aquest programa ha tingut un gran pes durant el desenvolupament. S'ha necessitat per crear totes les textures del joc, tant les empaquetades en els botons com les normals, com per exemple els ninotets que representen els jugadors.

7.2. DISSENY DE L'ARQUITECTURA

En el disseny de l'arquitectura s'hi troben tres elements que es relacionen entre ells per poder tenir la connexió desitjada en el mode online. El primer és la base de dades de Google, que és on s'emmagatzema tota la informació (com per exemple el que refereix als usuaris: id, avatar, nivell, etc). El segon, el servei de Google Games que és qui s'encarrega de fer totes les peticions i fer totes les comunicacions API client 1 <-> API client 2. I finalment el client que és tot usuari amb el joc fent peticions a través de l'API.



Il·lustració 9. Esquema de l'arquitectura del servidor

7.2.1. DISSENY DEL SERVIDOR

Tal i com s'ha explicat, pel joc s'utilitza l'API de Google Play Games. Amb aquesta eina podem fer un seguit d'operacions que es comenten a continuació:

7.2.1.1. INICIALITZACIÓ I CONFIGURACIÓ DE L'HABITACIÓ

Aquesta habitació virtual és l'espai on els jugadors es comuniquen. Google proporciona una llista d'usuaris, que són amistats teves en l'àmbit de Google (Google+[3]). Un client pot entrar de l'habitació si crea una partida nova o si s'uneix a una creada, i pot abandonar quan ho desitgi. En tot moment manté un conjunt d'usuaris d'habitació el qual actualitza cada vegada que un s'hi uneix o se'n surt.

Pots configurar-la de tal manera que tingui un límit de jugadors i que aquests siguin autoseleccionats o no.

7.2.1.2. GAMEPLAY I MISSATGERIA

Quan l'habitació és considerada plena, comença l'estat de joc i es pot utilitzar Google Play per difondre missatges entre els jugadors. A continuació un exemple del codi font que utilitza aquesta classe d'acció:

```
Games.RealTimeMultiplayer.sendReliableMessage(mGoogleApiClient, this,
                mMsgBuf, mRoomId,p.getParticipantId());
```

La variable *mMsgBuf* és la que té el valor en Bytes del missatge *p* és el jugador a qui se li envia. Scrammikub utilitza aquesta operació en bastantes ocasions. Aquestes són:

- *Master* difon el total dels jugadors
- *Master* difon les mans dels jugadors
- *Master* difon el pas d'un segon
- *Master* difon informació d'una màquina
- Jugador difon el seu moviment
- Jugador difon que ha agafat una bola de bossa
- Jugador difon que ha guanyat la ronda

7.2.2. DISSENY DEL CLIENT

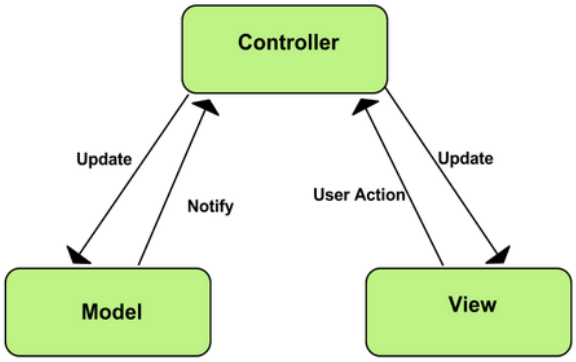
El client ha estat implementat utilitzant el framework LibGDX explicat anteriorment. Apart, en aquest punt, s'explica el partó utilitzat pel disseny del client i també la gestió de la memòria, que tal i com es veurà, està molt relacionat amb el framework.

7.2.2.1. PATRÓ MVC

El client segueix el patró d'arquitectura model-vista-controlador (MVC). Aquest patró és molt comú en el món de les aplicacions. És molt utilitzat en aquell software que les vistes

[3] Xarxa social de Google

tenen una gran importància o simplement en projectes que són grans que necessiten una estructura ben definida.



Il·lustració 10. Patró MVC [10]

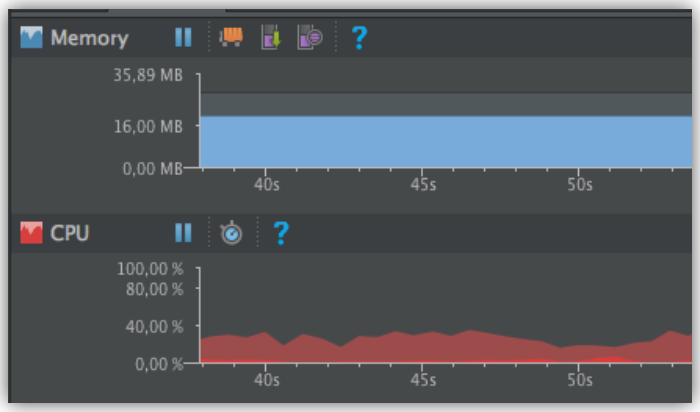
La vista és l'encarregada de mostrar a l'usuari la informació que requereixi, de manera interactiva i si es necessita. El model és qui gestiona les dades, representa la informació que opera tot el sistema i que la vista necessita mostrar a l'usuari. I el controlador invoca peticions de canvi d'informació del model quan la vista ho demana, o també li dóna aquella informació que necessita i que es troba en el model.

Utilitzant aquest patró s'assegura un codi de qualitat el qual és més comprensible i reutilitzable. De fet, utilitzant aquest patró s'està sacrificant eficiència a canvi d'un codi més estructurat i que permeti més fàcil manteniment, però es creu millor així donades les seves avantatges comentades.

7.2.2.2. ÚS DE LA MEMÒRIA

LibDGX ofereix una sèrie de mecanismes per facilitar la renderització de les imatges que es volen mostrar. Té unes classes anomenades actors, que són les que es portaran a pintar i s'hauran de renderitzar, per tant són aquelles que ocupen més memòria en el projecte. Aquests actors són les pantalles, botons, imatges, sons, vídeos, animacions, etc. En el cas d'Scrammikub el que es té és pantalles, imatges i botons.

Aquest framework dóna la responsabilitat a qui l'utilitzi de gestionar aquests actors. I per tant, si ja no es necessita algun d'aquests, manualment s'hauran d'alliberar per deixar de ser renderitzats i alliberar la memòria. En cas de no fer-ho s'estaria utilitzant memòria innecessària. Scrammikub té un control adequat respecte l'ús de la memòria ja que allibera aquells elements que no necessita cada vegada que canvia de context o s'hagi de deixar de renderitzar algun actor. Les següents imatges són captures del monitor de memòria d'Android Studio durant una partida i és veu l'anàlisi de memòria:



Il·lustració 11. Captura del monitor de memòria d'Android Studio

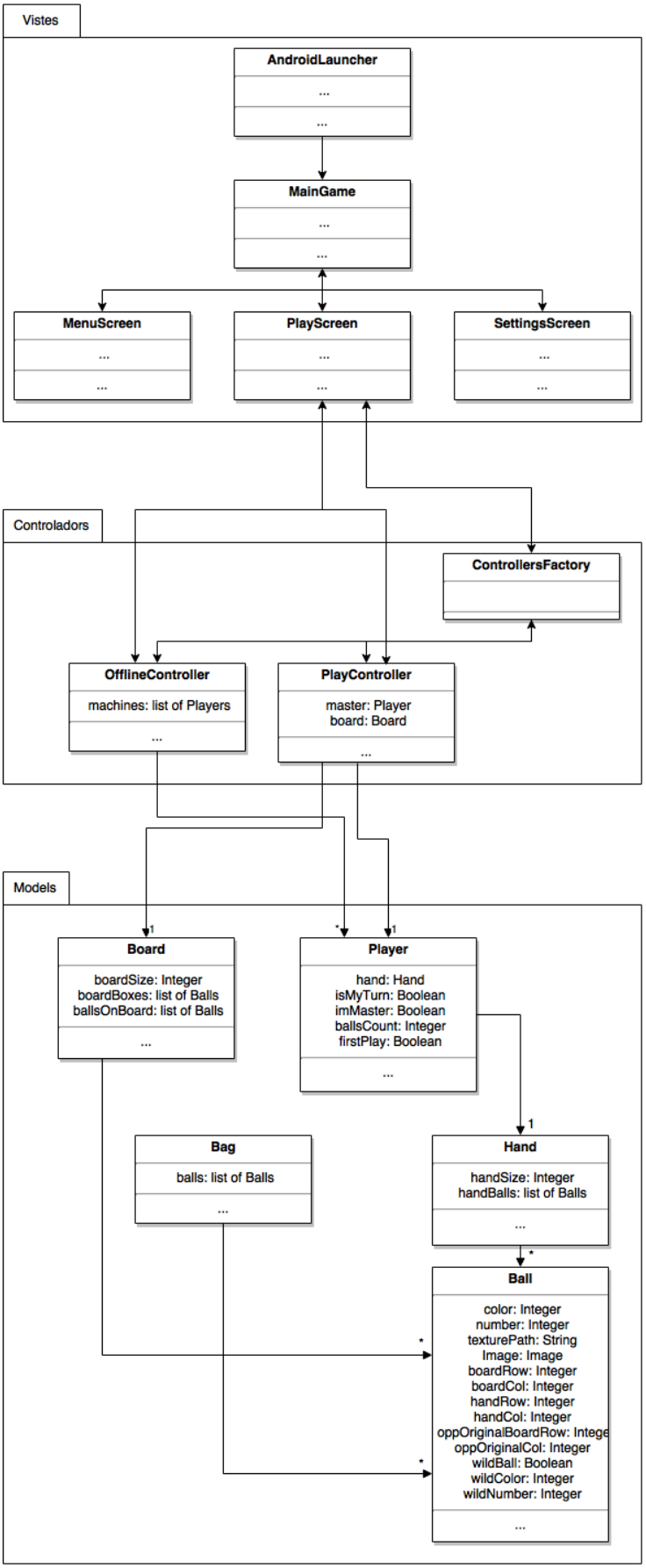
Aquesta imatge està presa durant la monitorització d'un smartphone Nexus 5 amb Android 6.0 en l'execució d'Scrammikub en un dels moments més crítics: una partida online amb jugadors online i màquines. S'observa que s'utilitza uns 18MB. Fent més proves es poden trobar valors d'entre 16MB a 26MB.

A més, per comparar els resultats amb una aplicació similar (Rummy[11]) s'ha instal·lat en el mateix smartphone un monitor del sistema (Simple System Monitor[12]). El monitor indica un increment de la memòria RAM del 6% executant Rummy i un increment del 4% executant Scrammikub.

Després d'aquest anàlisis i tenint en compte que es creu haver utilitzat eficientment, alliberant la renderització, tots els elements que ocupen en gran mesura la memòria, es conclou que Scrammikub té un ús de la memòria adequat.

7.3. CLASSES

Per al desenvolupament del projecte s'han utilitzat classes integrades dins del patró model-vista-controlador. Tot seguit es mostra el següent diagrama UML amb les classes relacionades entre elles i s'observa aquest patró:



Il·lustració 12. Diagrama de classes UML

Per entendre-ho millor, seguidament es descriuen totes aquestes classes.

7.3.1. ANDROIDLAUNCHER

Aquesta és la classe que es troba en el nivell més extern de tots, en executar el joc és la primera en prendre el control. A grans trets és l'adaptador o l'enllaç que lliga tot el disseny de domini i de presentació amb la capa de persistència proporcionada pels serveis de Google. En altres paraules, és la classe que maneja l'API de Google necessària per a la comunicació online. És el lloc on es poden executar totes les funcionalitats descrites en el *Disseny del servidor*.

El primer que fa aquesta classe és inicialitzar el joc inicialitzant una connexió amb els serveis de Google i també inicialitzant la resta de classes del disseny.

7.3.2. MAINGAME

AndroidLauncher és qui s'encarrega d'instanciar MainGame perquè es pugui donar pas a inicialitzar la resta de classes. Aquesta classe s'encarregarà bàsicament de crear les vistes, inicialitzar-les i donar pas a la visualització de la primera, MenuScreen.

7.3.3. MENUSCREEN

Aquesta és la classe encarregada d'ensenyar la vista inicial del menú i de gestionar totes les entrades que dóna l'usuari a la mateixa. Totes les entrades són a través de clicar botons:

ACCIÓ	BOTÓ	EXPLICACIÓ
Sortir del joc		Surt de l'app
Login		Fa LogIn
Logout		Fa LogOut
Modificar paràmetres		Va a la vista d'ajustaments
Veure invitacions		Mostra les invitacions
Jugar en mode Online		Mostra la llista d'usuaris
Jugar en mode Offline		Va a la vista de Play

Taula 8. Accions de l'usuari a la vista del menú inicial

7.3.4. SETTINGSSCREEN

Classe que gestiona la vista d'ajustaments i les entrades donades per l'usuari, les quals poden ser les següents:

ACCIÓ	BOTÓ	EXPLICACIÓ
Retornar		Torna a la vista del menú (guardant la configuració)
Augmentar/disminuir quantitat		Augmenta o disminueix la quantitat de robots/minuts/rondes

Taula 9. Accions de l'usuari a la vista d'ajustaments

7.3.5. PLAYSCREEN

Classe que gestiona la vista del joc durant una partida. És la classe realment principal del joc, on es centra tota l'activitat. Apart de controlar les accions de l'usuari, es comunicarà amb els controladors dels models per poder obtenir informació que necessiti l'usuari. Aquesta informació podria ser el resultat de comprovar si un moviment fet per l'usuari és o no correcte, el temps restant, els moviments duts a terme pels altres usuaris o màquines, etc. Les accions concretes que pot fer l'usuari són les següents:

ACCIÓ	BOTÓ	EXPLICACIÓ
Moure boles		Posicionament de les boles en el taulell
Passar el torn		Finalitzar i passar el torn
Agafar bola		Agafar bola de la bossa compartida pels jugadors
Desfer moviment		Desfer el moviment creat per l'usuari
Sortir de la partida		Sortir de la partida

Taula 10. Accions de l'usuari a la vista de partida

7.3.6. CONTROLLERSFACTORY

Aquesta classe és l'anomenada factoria dels controladors. Bàsicament s'encarrega d'instanciar tots els controladors (en aquest cas el *PlayController* i *OfflineController*) i

guardar-la. D'aquesta manera no es creen copies innecessàries dels controladors, sempre s'accedeix a la mateixa.

7.3.7. PLAYCONTROLLER

Controlador principal del joc, ja que és l'encarregat de gestionar la majoria d'operacions (modificar models, donar informació a la vista principal del joc, etc). D'aquesta classe es destaca que és on s'emmagatzema la informació del *Player* actual "*me*" i és on es gestiona el rellotge del joc. A més s'hi troba la *Bag* amb totes les Balls i la *Board* amb les posicions plenes o buides de Balls.

Aquest controlador té certes funcions[4] complexes que tot seguit s'expliquen amb detall.

7.3.7.1. UNDOBOARD

Aquesta és la funció que es crida quan l'usuari decideix desfer el seu moviment o quan se li ha acabat el temps sense clicar *Passar torn*.

A l'hora de desfer el moviment de l'usuari, aquesta funció ha de tenir en compte dos tipus de moviments: el de les boles que ha baixat de la mà al taulell i el de les boles que ja hi havia al taulell i l'usuari ha decidit moure.

PlayController va guardant aquests dos tipus de moviments durant la jugada del jugador i quan es crida a *undoBoard* comença per recórrer les boles baixades. Per cada bola que l'usuari ha baixat, la funció busca un espai buit a la seva *Hand* i li col·loca. Després continua amb el moviment de boles que ja eren al taulell tot cridant undoOpponentBallsMov que desfà el moviment.

7.3.7.2. UNDOOPPONENTBALLSMOV

En el procés de desfer el moviment se la crida. La solució d'aquesta funció no és tan trivial com *undoBall* ja que si es desfà el moviment en el mateix odre que l'usuari ha estat fent el seu moviment i el controlador enregistrant es podria donar casos en el que hi hagués bucles.

Il·lustració 13. Demo de

En aquesta imatge, si l'usuari mou la bola amb número 6 al lloc de la bola número 5 i la del número 5 al lloc de la del 6, quan la funció s'executi no podrà simplement agafar la bola en la posició que està ara la 6 i posar-la a la del 5 perquè quan vulgui desfer la del número 5 aquesta ja no hi serà, l'haurà sobreescrit.

[4] Codi complet amb totes les funcions annex al document

La solució a aquest problema és dóna a través d'eliminar una de les boles de la llista de moviments a desfer que s'està seguint (com si ja s'hagués desfet) i guardar-la en una llista auxiliar la qual després serà cridada per desfer-la.

7.3.7.3. MOVEOPPONENTBALLS

Aquesta funció, de manera similar a l'anterior, mou aquelles boles que els usuaris oponents han mogut en el seu torn. És similar a *undoOpponentBallsMov* perquè pot succeir el mateix cas dels bucles. La solució en aquest cas és la mateixa.

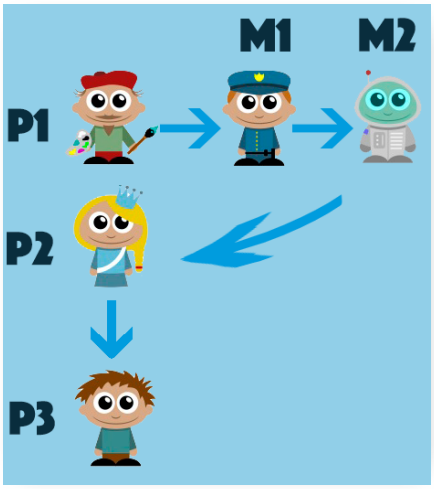
7.3.7.4. INITHANDSMASTER

Només l'usuari marcat com a *Master*[5] pot executar. De fet, el *Master* ho ha de fer per poder decidir quines boles tindran els jugadors a l'inici de la partida.

En aquest punt és important entendre el plantejament dels jugadors quan s'està jugant en mode online i hi ha màquines. S'ha dissenyat de tal manera que és el *Master* qui té el control d'aquestes màquines i per tant, la resta de jugadors online només sabran de la seva existència però no de la seva informació. La il·lustració 14 explica gràficament el flux d'una partida d'aquestes característiques:

Si P1, P2 i P3 són els jugadors online i M1 i M2 les màquines, durant la partida l'ordre dels jugadors serà P1, M1, M2, P2 i P3. D'aquesta manera, pels oponents del *Master* (o P1, en aquest cas) serà com si el torn de P1 s'allargués fins que acabi l'última màquina (M2).

Tot i això, per afavorir una millor experiència en el joc, els oponents podran veure com s'actualitzarà el torn (les boles i la senyal vermella darrera els ninotets vermella que indica el torn) per cada màquina.



Il·lustració 14. Flux d'una partida amb jugadors online i màquines

Entès el flux de joc, s'entén que *initHandMaster* inicia totes les *Hands* del joc (de jugadors online i offline), les envia als jugadors online perquè les inicialitzin i les inicialitza a les màquines, donat que és ell el *Master* i s'encarrega de gestionar-les.

[5] Usuari encarregat de crear la partida en el mode online. Se li associa el comptador del temps (rellotge), les màquines i és qui inicia les *Hands* dels usuaris aleatòriament.

7.3.7.5. CORRECTMOVEMENT

Tal i com indica el nom és una funció encarregada de comprovar si un moviment és o no correcte. En aquesta funció queden guardades la majoria de les normes del joc, així com que s'han de fer sèries del mateix color i diferent numero incremental o mateix número i color estrictament diferent (vistes en la *Normativa del joc*).

Sense entrar en profunditat, aquesta funció ressegueix tota *Ball* que es trobi a *Board* i comprova si està ben posicionada. Aquest últim criteri ve donat per si la bola es troba dins d'una línia (seqüència de boles vertical o horitzontal) que és correcte. Aquest últim criteri ve donat quan la línia té almenys 3 boles i són d'un dels dos tipus comentats, anomenats *group set* els que són tots del mateix número i color diferent i *run set* els que són tots del mateix color i número incremental.

Si troba una línia que no està bé, la bola que estava comprovant és per tant incorrecte. Para de resseguir les boles i informa d'un moviment incorrecte.

Aquesta funció s'utilitza cada vegada que l'usuari *Passar torn.* En el cas que el moviment sigui correcte, i hagi fet algun moviment, passa de torn. En cas contrari, serà informat i haurà de canviar-lo.

7.3.7.6. CALCULATECOLNUMWILD

Aquesta és l'operació que s'encarrega de traduir una *Wild Ball* o un comodí. S'utilitza quan en operacions com l'anterior en la que s'havia de comprovar bola per bola si era correcte o no, la qual havia de comprovar també tota la seva línia. És en aquestes operacions quan necessitem saber quin valor (color i número) ha volgut donar l'usuari en el comodí havent-lo posat on la posat.

Aquesta és una funció complexa la qual necessita llegir l'estat actual de les boles que té al costat per encertar quin valor té. Comença comprovant si està en una línia vertical o horitzontal a través de funcions individuals que apart de comprovar-ho, li retornen el valor del comodí en cas de ser-ho. Aquestes funcions recorren la línia en busca de saber quin tipus de conjunt foment (*group* o *run*) i depenent dels seus valors provaran si el valor que hauria de tenir el comodí realment és un *correctMovement*.

7.3.8. OFFLINECONTROLLER

Aquesta classe és el controlador de gestionar les màquines. En altres paraules, és el mòdul d'intel·ligència artificial del programa. Té com a objectiu imitar el comportament

humà, tal i com si estigués jugant. Per aconseguir-ho es necessari que aquest controlador sàpiga trobar, amb les boles que disposa cada màquina, un moviment vàlid i mostrar-lo. I per suposat, aquesta classe guarda totes les màquines, com a conjunt de *Players*.

Aquest mòdul, tal i com es va acordar amb el director del projecte, és una primera implementació que dóna pas a millores futures. Scrammikub doncs, les màquines offline que ofereix només busquen possibles moviments amb les boles que disposa a la seva *Hand*. Això significa que les màquines tenen la capacitat de construir i baixar al taulell conjunts correctes de boles en espais lliures i també col·locar correctament boles de la seva Hand en línies del taulell que no tenen cap comodí. Per altre banda, una màquina no sap reorganitzar boles del taulell per poder afavorir baixar més boles.

OfflineController ofereix unes màquines que són intel·ligents però no en el grau suficientment com per guanyar a una persona humana, almenys la majoria de vegades. A continuació es descriuen aquelles funcions del controlar que són més complexes i tenen un paper i un pes més rellevant:

7.3.8.1. NEXTOPPMOV

Un cop *Master* acaba el seu torn, tal i com segueix el flux explicat, li toca el torn de les màquines (si és que l'usuari ho ha configurat així). El sistema està pensat de manera que cada màquina tardi un temps definit, per donar realisme. *NextOppMov* es crida per crear el moviment de la màquina però no és fins que ha passat aquest temps determinat que no es mostra a la resta d'usuaris i al propi *Master*.

7.3.8.2. MAKEOFFLINEMOVEMENT

Aquesta funció la crida l'anterior per decidir el moviment de la màquina que li toqui. És el cervell de les màquines intel·ligents. Es divideix en dues parts, formar conjunts i utilitzar boles individuals.

En la primera part, el que primer es necessita és buscar tots els conjunts possibles amb les boles que té a la mà la màquina. Això ho fa amb una funció anomenada *lookForSets* i s'explica en el següent punt. Donada aquesta llista, comprova que si és el primer torn la puntuació del moviment sigui igual o major que 30 i que hi hagi espai lliure per poder-lo aplicar (*cooFreeBoardBox*), sent així prepara el canvi perquè quan sigui el temps es mostri a l'usuari.

En la segona, es crida a una funció encarregada de retornar totes les línies del taulell per poder-la resseguir i així buscar per cada una si posar una de les boles que es té a *Hand*

a un extrem de la línia és correcte o no. Sent correcte, es guardaria com a moviment a mostrar a l'usuari de la mateixa manera que la primera part.

7.3.8.3. LOOKFORSETS

El que fa aquesta és resseguir les boles que se li passen i per cada una comprova si forma conjunt *group o run* amb alguna de les altres boles de la mà. Va ampliant la línia a mesura que va trobant boles a la mà que formin línia. Un cop les ha mirat totes, si amb aquesta bola en qüestió ha trobat una línia possible, l'afegeix a una llista auxiliar i continua.

7.3.9. PLAYER

Classe model que té tota la informació d'un usuari/jugador. Guarda informació com la seva *Hand*, si és el seu torn o no, si és *Master* o la id. Els controladors són els encarregats d'obtenir i modificar la seva informació tal i com el patró model-vista-controlador estableix.

7.3.10. BOARD

Classe model del taulell del joc. Tots els jugadors comparteixen la mateixa informació respecte aquesta classe. En ella s'hi guarda la grandària del taulell i les boles que hi té en ell. Igual que la resta de models, són els controladors qui poden accedir i modificar-la.

7.3.11. BAG

Classe model que conté totes les boles possibles del joc. Depenent del nombre de jugadors en tindrà més o menys boles, tal i com s'estipula a les *Normes del joc*. *Bag* guarda totes aquestes boles i té funcions que li permeten inicialitzar-la, obtenir boles i generar conjunts de posicions aleatòries de la bossa (necessari per inicialitzar les *Hands* dels jugadors.

7.3.12. HAND

Classe model que guarda totes les boles que té un jugador a la seva mà. Apart, també permet el control de les posicions lliures o ocupades de la mà, necessari per a altres operacions dels controladors.

7.3.13. BALL

Classe model base del joc. Tot el joc gira entorn a aquesta classe. Emmagatzema la informació de les boles tals com el número i el color i si és o no un comodí. A més, també

guarda les coordenades dins el taulell i la *Hand* i facilita la seva textura i imatge que es mostra a la vista.

8. IMPLEMENTACIÓ

8.1. TASQUES D'SCRAMMIKUB

A la Planificació temporal estan descrites aquelles tasques que abans de desenvolupar el sistema es preveien implementar. A continuació es mostra la llista completa de les que s'ha seguit per desenvolupar Scrammikub i com s'han dut a terme:

8.1.1. FER LOGIN / LOGOUT

El repte era tenir una gestió d'usuaris amb la que es pogués connectar-los entre ells perquè poguessin buscar-se, enviar-se invitacions i enviar-se missatges durant el joc. Un cop es va optar per Google Play Games, la complicació de fer Login o Logout només va ser la d'instal·lar l'API oferida per Google i la d'integrar un botó que lligués la pulsació amb l'acció de login o logout de l'API.

Per poder fer aquestes accions i algunes de les que es veuran a continuació, va ser necessari crear una pantalla per a que l'usuari pogués interactuar i fer tals accions. S'anomena *Menu*.

Apart del codi, a la consola de Google es va haver de configurar de tal manera que permetés la connexió.

8.1.2. SORTIR DE L'APLICACIÓ

Es va afegir un botó que conduís al tancament de l'aplicació mitjançant una crida de LibGDX que se n'encarrega.

8.1.3. CONFIGURAR PARÀMETRES

El joc té un límit mínim de 2 jugadors i un de màxim de 6 jugadors, té l'opció que el torn duri 1, 2 o 3 minuts i es pot jugar al millor d'una sèrie de rondes. Es necessitava doncs un espai diferent al *Menu* on l'usuari pogués modificar aquests paràmetres i es va crear una pantalla anomenada *Settings.* A més, es van definir unes variables globals per emmagatzemar el valor d'aquests paràmetres per a poder ser consultats en qualsevol lloc del joc.

8.1.4. INVITAR A JUGADORS ONLINE I VEURE INVITACIONS

De la mateixa manera que el LogIn i el LogOut simplement va fer falta lligar un botó a una acció de l'API que ho facilitava.

8.1.5. TOTS ELS JUGADORS COMPARTIM BOSSA

La bossa és on es guarda totes les boles del joc. És un element que necessitava ser actualitzat cada vegada que un jugador agafava una bola i a més havia de fer-se de manera totalment aleatòria. Es va haver d'implementar de tal manera que totes les bosses dels jugadors fossin igual i quan algú agafés una bola transmetés a la resta de jugadors l'índex de la bossa que havia agafat per poder marcar com a buida la posició d'aquella bola.

A més a més, aquesta bossa té una grandària diferent depenent del nombre de jugadors tal i com s'explica en la *Normativa del joc* amb el nombre de sèries de colors necessàries.

8.1.6. POSICIONAR BOLES AL TAULELL

L'acció principal del joc transcorre en el taulell i una de les funcionalitats bàsiques és la de posar boles de la mà al taulell. El sistema emmagatzema per una banda les boles que té el jugador a la seva mà i per altre les boles que hi ha damunt el taulell. Per resoldre aquest posicionament es va optar per actualitzar les dues fonts i a la vegada avisar als altres jugadors de les boles posades al taulell.

Aquesta i algunes de les següents tasques es donen a una pantalla nova anomenada *Play* que és on el jugador desenvolupa tot el joc i tots els jugadors interaccionen entres si.

8.1.7. MOURE LES BOLES DEL TAULELL

El jugador havia de poder moure les boles que els seus oponents (o ell mateix en torns previs) havien col·locat en el taulell. Es va aconseguir tractant les boles que ja hi havia al taulell un cop començava el torn, d'una manera diferent a les que es posaven en el mateix. D'aquesta manera es podia saber quines boles es podien tornar a la mà o no i es podia moure lliurement les boles del taulell.

8.1.8. DESFER UNA BOLA

Per donar una millor experiència de joc als jugadors, es va creure necessari l'opció de tornar a la mà una bola concreta que s'havia posicionat al taulell. Per fer-ho es va implementar la funcionalitat de que si una bola era tocada dues vegades seguides, aquesta, si no era una bola que ja hi havia abans de començar el torn, tornava a la mà.

8.1.9. DESFER EL MOVIMENT COMPLET

Desfer el moviment complet també es va pensar per afavorir una millor experiència a l'usuari però a més, era necessari per quan un jugador se li acabava el temps del seu

torn ja que se li havia de desfer automàticament el seu moviment. Es va resoldre amb la funció *undoBoard* explicada anteriorment.

8.1.10.PASSAR TORN

Quan l'usuari ha acabat els seus moviments o quan se li ha acabat el temps s'ha de passar de torn. Scrammikub ho pot fer a través d'una funció específica i un botó que està lligat a aquesta funció.

8.1.11.MOVIMENT CONTROLAT, ÉS CORRECTE O NO

Si s'intenta passar el torn manualment, el sistema ha de comprovar si el moviment que proposa l'usuari és o no correcte ja que si no ho és no li deixarà fer-lo ni li deixarà passar el torn. Aquest control es va resoldre amb la funció *correctMovement* explicada anteriorment.

8.1.12.AGAFAR UNA BOLA DE LA BOSSA COMPARTIDA

Aquesta és una acció necessària i bàsica en la mecànica i normativa del joc. S'ha implementat a través d'un botó específic que fa la mateixa acció de quan a un usuari se li acaba el temps i no ha agafat cap bola prèviament. El que fa aquesta acció és informar de l'índex de la bola agafada a la resta de jugadors.

8.1.13.JUGAR AMB COMODINS

Gràcies als comodins, el joc cobra una nova dimensió. Amb ells pots donar molt de joc i inventar possibles moviments. Són clau, doncs, pel joc. La implementació i solució a aquesta tasca està donada amb la identificació de les *Ball* (marcades com a comodí) i a través de la traducció de les mateixes amb la funció explicada *calculateColNumWild*.

8.1.14.SORTIR DEL JOC

Durant el joc o quan s'acaba el mateix, el jugador ha de poder tornar a la pantalla *Menu* per decidir noves accions. A la pantalla *Play* hi ha un botó que tanca la pantalla i et porta a *Menu* i també si s'acaba el joc, opera la mateixa funció.

8.1.15.CONTROLAT PEL TEMPS

L'element del temps és un factor clau en el joc. S'ha implementat un rellotge que es sincronitza amb el de tots els jugadors a cada segon. Aquest rellotge el controla el jugador *Master* i assegura que els jugadors facin els moviments dins el seu temps.

8.1.16.SER AVISAT DEL QUE PASSA PEL SISTEMA

És important que un jugador pugui saber en tot moment què està succeint i si per exemple un moviment que està fent no és correcte, hauria de saber que no s'efectua per aquest motiu. S'ha establert una zona a la part superior dreta de *Play* per avisar d'aquests estats. A continuació podem veure una llista dels missatges que es donen i quan es donen:

MISSATGE	MOMENT
YOU ALREADY GOT ONE	Quan un jugador ja ha agafat una bola i n'intenta agafar una altre
GAME FINISHED	Quan algun jugador de l'habitació virtual s'ha desconnectat
INIT MELD VALUE IS 30 OR MORE	Quan es fa el primer moviment de la partida però aquest malgrat ser correcte no té un valor de 30 o més
MAKE AN INITIAL MELD OR DRAW A BALL	Si un jugador passa de torn però no ha baixat cap bola ni n'ha agafat cap
NOT YOUR TURN	Quan un jugador intenta fer alguna acció però no és el seu torn
NOTHING TO UNDO	Quan un jugador intenta desfer el seu moviment però no té res a desfer
WAITING TURN...	Quan un jugador ha acabat el seu torn

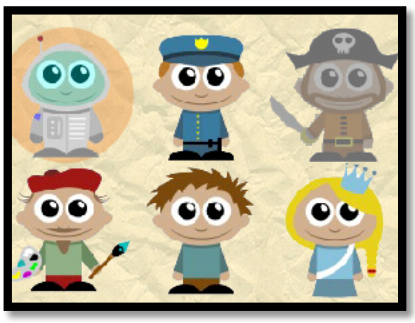
Taula 11. Missatges del sistema

8.1.17.VEURE L'ESTAT DE LES RONDES

El joc permet jugar a rondes i és important que els jugadors sàpiguen en quina ronda estan. A sobre dels missatges del sistema s'ha posat un text indicatiu de quantes rondes es porta i quantes són en total.

8.1.18.VEURE L'ESTAT DELS JUGADORS, ACTIVITAT I TORN

Els jugadors necessiten saber de qui és el torn actual, altrament estarien desconnectats del que està passant en la partida. S'ha creat unes textures especials per indicar de qui és el torn i quants jugadors hi ha actius. S'ha creat un mecanisme encarregat de transmetre qui és el nou jugador que té el torn a tots els jugadors. Bàsicament, quan un jugador passa el torn ho informa a la resta. A continuació una imatge que mostra la textura comentada, en gris els jugadors no actius i en vermell el qui té el torn:



Il·lustració 15. Textura de l'estat dels jugadors

8.1.19.VEURE L'ESTAT DEL TEMPS

L'estat del temps és una informació molt necessària si es vol "connectar" el jugador en el joc, de la mateixa manera que l'estat dels jugadors. S'ha creat una barra verda que disminueix cada segon que passa, estiguis en el teu torn o no, i que torna al l'estat inicial quan es passa el torn. A continuació una imatge que ho mostra:

Il·lustració 16. Textura de l'estat del temps

8.1.20.VEURE LA FINALITZACIÓ DE LA PARTIDA

El jugador ha de ser informat de si ha guanyat o perdut una ronda o el joc. S'ha dissenyat unes textures per ser mostrades quan algun jugador guanya o perd:

Il·lustració 17. Textura de ronda acabada

8.1.21.SISTEMA TANCAT, SENSE ERRORS

Tot el sistema ha d'oferir una experiència a l'usuari sense cap error, com podria ser un tancament inesperat del joc o que el sistema no segueixi la normativa pròpia del joc. Per garantir aquest sistema tancat s'ha fet proves de cada una de les tasques. Una tasca no era correcte si no funcionava perfectament ella i les anteriors implementades. Bàsicament s'ha seguit la metodologia SCRUM explicada.

8.1.22.ESTÈTICA

El projecte pretenia tenir una estètica mínima per poder disposar de totes les funcionalitats i, en cas de tenir-ho, es milloraria. Així podria millorar l'experiència del jugador. S'ha pogut implementar totes les funcionalitats que es requeria així que s'ha creat un entorn estètic més allà del mínim tal i com s'observa amb els botons i les textures del joc.

8.1.23.JUGAR EN MODE ONLINE I OFFLINE

El joc tenia el requisit inicial de poder jugar-se en mode online i offline. Online contra altres jugadors humans i offline contra jugadors artificials del sistema. L'elecció del mode s'ha implementat en la pantalla *Menu* amb botons diferents.

Per aconseguir el mode online s'ha utilitzat l'API de Google per la comunicació i conseqüent implementació. I per el mode offline s'ha integrat un mòdul d'intel·ligència artificial que simula a jugadors humans.

9. IDENTIFICACIÓ DE LLEIS I REGULACIONS

Durant el desenvolupament del projecte no s'ha pogut deixar de banda el factor de les lleis i regulacions, ja que podrien ser un factor condicionant al resultats.

S'ha identificat que en relació a Scrammikub hi ha una relació indirecta amb una llei que protegeix les dades dels usuaris. A més, hi ha certes llicències de productes que s'han utilitzat que necessiten ser explicades.

9.1. TRACTAMENT DE LES DADES DELS USUARIS

Tot sistema que emmagatzemi informació sensible d'usuaris està regida per la llei orgànica de protecció de dades i també de la llei antipirateria. Aquesta llei especifica que s'ha de complir que els usuaris estiguin informats del recolliment d'informació seva personal i com serà utilitzada, ha d'implementar la seguretat necessària per protegir la informació i ha de registrar les dades en l'Agència Espanyola de Protecció de Dades.

Per altre banda, Scrammikub no és un sistema que emmagatzemi cap tipus d'informació dels usuaris. Tot el tema relacionat amb aquesta informació la tracta Google amb els seus serveis. Són els usuaris que han de consentir que Google administri les seves dades, i això ho fan quan es creen una compta de Google o un usuari per jugar amb Google Play Games (sigui pel joc que sigui). Per tant, el joc desenvolupat no té cap restricció de cap llei.

9.2. SOFTWARE LLIURE

Tot el software utilitzat en aquest projecte és de tipus privatiu de propietat de diferent empreses. LibGDX és una excepció, és software lliure i està sota la llicència Apache 2 per treballar-hi i sota Creative Commons 3 pels jocs desenvolupats.

Apache 2 [13]

Permet a l'usuari utilitzar el software amb la llibertat de fer-lo servir amb el propòsit que sigui, distribuir-lo, modificar-lo i distribuir versions modificades del que s'estigui creant. Aquesta llicència, a diferència d'altres, no exigeix que els productes que es creen siguin sota la mateixa llicència.

Creative Commons 3 [14]

El producte ha de complir les següents condicions:

- Reconèixer l'autor
- De tipus no comercial
- Sense opció d'obres derivades a no ser que siguin sota la mateixa llicència

10. RESULTATS

A continuació es mostra un seguit d'il·lustracions que ensenyen el resultat del projecte, Scrammikub acabat. Es relaciona cada imatge amb les tasques que s'han implementat (les funcionalitats del programa).

Pantalla *Menu*

Il·lustració 18. Resultat: pantalla Menu

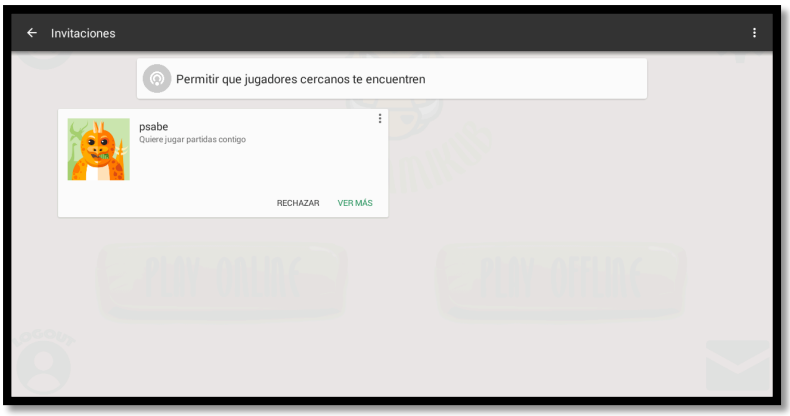
Aquesta és la pantalla on es pot fer Login/LogOut, sortir de l'aplicació (8.1.2), anar a la configuració dels paràmetres, jugar a les dues modalitats de joc i veure les invitacions.

Veure invitacions

Il·lustració 19. Resultat: veure invitacions

Aquesta és la finestra que apareix en fer Login (8.1.1).

Veure invitacions



Il·lustració 20. Resultat: veure invitacions

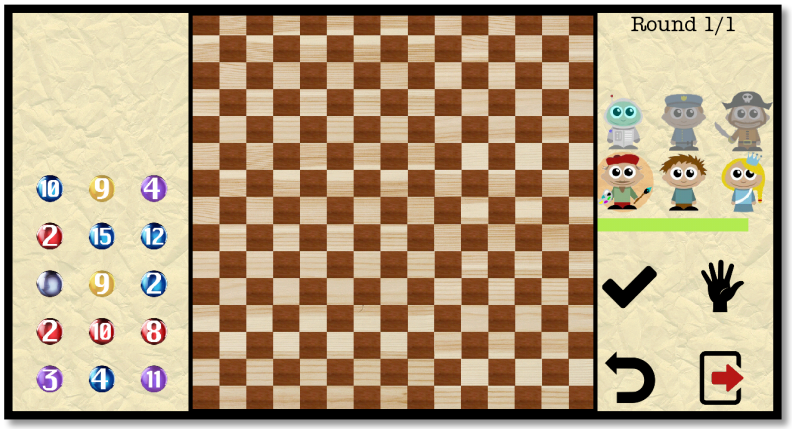
Finestra que s'obre quan vols veure les invitacions que t'han fet (8.1.4).

Pantalla *Settings*

Il·lustració 21. Resultat: pantalla Settings

Pantalla *Settings* amb totes les seves configuracions possibles (8.1.3).

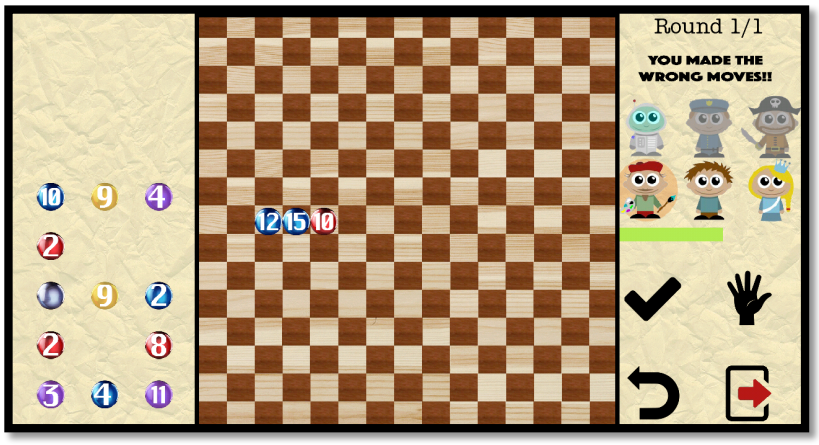
Pantalla *Play*



Il·lustració 22. Resultat: pantalla Play

Pantalla Play amb totes les opcions descrites (8.1.5-8.20).

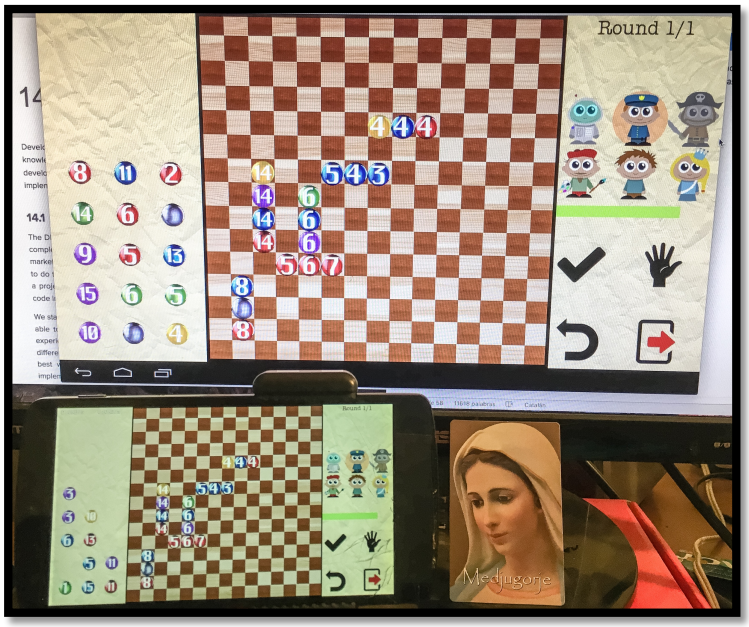
Moviment incorrecte



Il·lustració 23. Resultat: moviment incorrecte

Moment en el que es prem el botó de passar torn, el sistema detecta que el moviment realitzat és incorrecte i ho mostra a l'usuari (8.1.10, 8.1.11, 8.1.16).

Partida online



Il·lustració 24. Resultat: partida online

Execució d'una partida online. Els jugadors són: l'usuari amb el mòbil, l'usuari amb l'emulador de l'ordinador i 3 màquines (8.1.23)

10.1. TREBALLS FUTURS

A continuació es mostra una llista de les possibles funcionalitats d'ampliació. Aquelles opcions que en un futur se l'hi podria implementar a Scrammikub.

- Millorar l'entorn estètic
- Millorar jugabilitat i l'experiència de joc
    - Les boles s'arrosseguen en comptes d'haver-se de clicar per seleccionar la bola i clicar per seleccionar el lloc de dipositar-la
    - Visualització de la bossa amb les boles
    - Visualització de jugadors agafant bola
    - Boles movibles en la zona on estan les boles de *Hand*
    - Visualització de quan falten pocs segons per acabar el torn especial
    - Desfer l'últim moviment (no només tot sencer)
- Màquines més intel·ligents
- Afegir paràmetre d'intel·ligència (nivell de les màquines)
- Flux de torns aleatori, evitant que sigui estrictament: *Master* -> màquines -> jugadors online
- Afegir sons i música de fons
- Millorar l'opció de rondes. Que en cas d'empat guanyi el que tingui un valor menor al recompte
- Major rapidesa quan es mou entre pantalles
- Popup d'invitacions
- Integració del joc a la plataforma iOS
- Taula de classificació. Taula del millor del món i taula del millor dels teus amics
- Opció de revenja
- Opció de demanar pista al sistema si es va perdut

11. CONCLUSIONS

Aquest projecte ha aconseguit que es desenvolupés Scrammikub, però més important encara, ha servit per posar en pràctica tots els coneixements teòrics que s'han donat en el grau, inclosa la part de la planificació, seguir una metodologia de treball i escollir la tecnologia adequada.

S'ha donat solució a un problema inicial, es tenia la necessitat de desenvolupar un joc de taula amb unes normes determinades per a dispositius mòbils. S'ha contextualitzat, s'ha investigat el mercat actual, definit l'abast el qual es volia arribar, planificat, estudiat econòmicament i també a nivell de sostenibilitat i finalment s'ha desenvolupat el sistema amb les eines més optimes i seguint les tasques necessàries.

Seguint els objectius que s'havien posat a l'inici del projecte, s'ha identificat com a perfil d'usuari a tot jugador de jocs de taula que utilitza dispositius mòbils, s'ha dissenyat un software de qualitat per satisfer aquest perfil d'usuari i finalment s'ha desenvolupat. Tal i com ja es va preveure, la complexitat de resoldre el problema ha estat la multiconnexió d'usuaris i el mòdul d'intel·ligència artificial. Malgrat això, el resultat compleix les expectatives inicials.

Aquest treball ha permès experimentar què és dur a terme un projecte de creació de software de principi (planificació) a fi (desenvolupament i testing). A través de la metodologia de treball àgil SCRUM, s'ha après a treballar dividint el pes del treball en tasques o històries d'usuari. Ha sigut fàcil saber en tot moment en quina quantitat de software havíem desenvolupat ja que totes les tasques tenien el seu pes i podies saber quan treball havies fet i quan et quedava per fer.

L'Scrammikub dissenyat a través d'aquest projecte no deixa de ser una primera versió del joc que pot donar peu a implementacions futures que el facin més atractiu i amb més opcions de joc. I el següent pas seria sens dubte el de millorar el mòdul intel·ligent, que sense menysprear l'actual, no deixa de ser una primera implementació amb visió de ser millorada.

Finalment, pel que fa a les competències tècniques, tot seguit és dóna resposta a si s'han assolit o no:

CES1.1: *Desenvolupar, mantenir i avaluar sistemes i serveis software complexos i/o crítics.* [En profunditat]

S'ha desenvolupat, amb opcions de manteniment futur, i avaluat un sistema complex: un joc de taula amb modes online i offline.

CES1.2: *Donar solució a problemes d'integració en funció de les estratègies, dels estàndards i de les tecnologies disponibles.* [Una mica]

S'ha escollit el disseny i l'arquitectura més òptima per al desenvolupament d'Scrammikub.

CES1.4: *Desenvolupar, mantenir i avaluar serveis i aplicacions distribuïdes amb suport de xarxa.* [En profunditat]

El projecte inclou la implementació de l'API de Google Play Games que dóna la possibilitat de connectar els jugadors entre si de manera online, amb suport de xarxa.

CES1.7: *Controlar la qualitat i dissenyar proves en la producció de software.* [Bastant]

S'ha seguit la metodologia SCRUM que assegura un producte de qualitat a través de proves en la producció del software en cada sprint (conjunt de tasques en aquest cas, o funcionalitats).

CES1.8: *Desenvolupar, mantenir i avaluar sistemes de control i de temps real.* [Una mica]

El sistema desenvolupat és a temps real, on diferents usuaris estan comunicats entre si en el mateix moment.

CES2.1: *Definir i gestionar els requisits d'un sistema software.* [Una mica]

El sistema no s'ha desenvolupat sense abans haver fet una bona documentació de tots els elements necessaris. D'aquesta manera s'ha definit i gestionat els requisits d'Scrammkub abans de ser desenvolupat.

CES3.1: *Desenvolupar serveis i aplicacions multimèdia.* [En profunditat]

Scrammikub és un joc que interactua amb l'usuari a través d'informació representada amb textures gràfiques i text. És doncs una aplicació capaç d'entretenir a usuaris gràcies a la mecànica del joc i el seu entorn multimèdia.

12. REFERÈNCIES

[1] Pàgina web de Statista, "Number of smartphone users worldwide…" [Online]. Disponible a: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ [Consulta: setembre 2016]

[2] Pàgina web de Blogspot, Imatge [Online]. Disponible a: http://1.bp.blogspot.com/-jFtwdSfX4sw/Uu7BtvJSQ-I/AAAAAAAAC7U/zcHT_ixw1-E/s1600/rummikub.gif [Consulta: setembre 2016]

[3] Pàgina web de Wikimedia, Imatge [Online]. Disponible a: https://upload.wikimedia.org/wikipedia/commons/5/5d/Scrabble_game_in_progress.jpg [Consulta: setembre 2016]

[4] Pàgina web de Google Play, "Aplicacions Android" [Online]. Disponible a: https://play.google.com/store/search?q&c=apps [Consulta: setembre 2016]

[5] Pàgina web de Apple, "iTunes en un vistazo" [Online]. Disponible a: https://itunes.apple.com/es/genre/ios/id36?mt=8 [Consulta: setembre 2016]

[6] Pàgina web de iphoneworld, "Quién tiene más aplicaciones..." [Online]. Disponible a: http://www.iphoneworld.com.es/2015/01/quien-tiene-mas-aplicaciones-apple-app.html [Consulta: Octubre 2016]

[7] Pàgina web de Google Play, "Rummikub" [Online]. Disponible a: https://play.google.com/store/apps/details?id=com.kimaia.rummikub [Consulta: setembre 2016]

[8] Pàgina web de Git, "git" [Online]. Disponible a: https://git-scm.com [Consulta: setembre 2016]

[9] Pàgina web de Google Play Games, "Google Play Games Services" [Online]. Disponible a: https://developers.google.com/games/services/ [Consulta: desembre 2016]

[10] Pàgina web de Google, "MVC Architecture" Imatge [Online]. Disponible a: https://developer.chrome.com/apps/app_frameworks [Consulta: gener 2017]

[11] Pàgina web de Google Play, "Rummy – Offline" [Online]. Disponible a: https://play.google.com/store/apps/details?id=com.sngict.rummy [Consulta: gener 2017]

[12] Pàgina web de Google Play, “Simple System Monitor” [Online]. Disponible a:
https://play.google.com/store/apps/details?id=com.dp.sysmonitor.app&hl=es [Consulta: gener 2017]

[13] Pàgina web de Apache, “Apache” [Online]. Disponible a:
https://www.apache.org/licenses/LICENSE-2.0 [Consulta: gener 2017]

[14] Pàgina web de Creative Commons, “Licencias” [Online]. Disponible a:
http://es.creativecommons.org/blog/licencias/ [Consulta: gener 2017]

13. ANNEX

13.1. COM INSTAL·LAR SCRAMMIKUB

Scrammikub es troba adjunt al document en forma d'executable .apk. Per instal·lar-lo només es necessita copiar aquest arxiu en el dispositiu Android i executar-lo des d'un explorador d'arxius. Després d'instal·lar-lo ja es podrà fer servir.

Malgrat això, es necessita també tenir instal·lat Google Play Game Services pel mode online. La majoria d'Smartphones ja ho tenen instal·lat i en cas contrari es pot trobar fàcilment en el mercat d'aplicacions Google Play. A més, també és necessari tenir una compta a aquest servei, i es pot crear des de la mateixa aplicació o des del joc quan s'intenta fer LogIn.

13.2. COM IMPLEMENTAR VERSIONS FUTURES

Adjunt a aquest treball hi ha el repositori del projecte. Per implementar noves versions del joc seria necessari integrar el projecte a Android Studio. A més, a fi de lligar la consola de Google amb l'API del projecte, s'hauria d'enllaçar el joc a la consola del desenvolupador nou i afegir la identificació en el codi (arxiu ids.xml).

En la documentació de les mostres que ofereix Google de jocs utilitzant aquesta API es pot trobar la següent informació que serien els mateixos passos a seguir però per a aquest projecte.

<h2>How to run a sample</h2>

1. Set up the project in Developer Console. For more info:

    https://developers.google.com/games/services/console/enabling

  Note your package name and the APP ID of the project.

1. Create leaderboards/achievements as appropriate for the sample (see the ones that the sample needs in its res/values/ids.xml). You can do this automatically by clicking the link below for the sample you want to configure:
  1. [Type a Number](http://playgameservices.github.io/android-basic-samples/config-magic/index.html?sample=typeanumber)
  1. [Trivial Quest](http://playgameservices.github.io/android-basic-samples/config-magic/index.html?sample=trivialquest)

Pick a set of instructions below depending on whether you're using Eclipse or Android Studio.

<h3>If you're using Eclipse...</h3>

1. From the command line run Scripts/make_eclipse_compat (or Scripts/make_eclipse_compat.cmd on Windows). This creates the directory structure needed to import the projects correctly.
1. Start Eclipse
1. Import the Google Play Services library project (available for download through the SDK manager):
  1. Click **File | Import | Android | Existing Android Code into Workspace**
  1. Select `SDK/extras/google/libproject/google_play_services/google_play_services_lib` (where `SDK` stands for the path where you installed your Android SDK)
  1. Click **Finish**
1. Import `eclipse_compat/libraries/BaseGameUtils` **as a library**

   1. Click **File | Import | Android | Existing Android Code into Workspace**
   1. Select the `eclipse_compat/libraries/BaseGameUtils` project
   1. Click **Finish**
   1. Right-click on `BaseGameUtils`, then click **Properties**
   1. In the project properties window, click the **Android** section
   1. Check the **Is Library** checkbox
   1. Add a reference to the `google_play_services_lib` project (click **Remove** to remove any broken references, then click **Add** to add the correct one)
1. Import the desired sample from the `eclipse_compat` directory (Project | Import | Android | Existing Android Source)
1. Go into the project properties window for that project (right-click, **Properties**) and check that this project has a reference to the `BaseGameUtils` project.

Your project should now compile. However, don't run it yet, since you still need to adjust your game's IDs
in order for the sample(s) to work.

Now jump to the *Modify IDs, compile and run* section and continue to follow the instructions there.

<h3>If you're using Android Studio...</h3>

1. Open Android Studio and launch the Android SDK manager from it (Tools | Android | SDK Manager)
1. Check that these two components are installed and updated to the latest version. Install or upgrade
   them if necessary.
   1. *Google Play Services*
   1. *Google Repository*
1. Return to Android Studio and select *Import Project*
1. Select the **BasicSamples** directory
1. Select "Import from existing model - Gradle"

<h3>Modify IDs, compile and run</h3>

To set up a sample:

1. Change the application id in the build.gradle file to your own package name
   (the same one you registered in Developer Console!). You will have to update
   the build.gradle file for each sample you want to run. There is no need to
   edit the AndroidManifest.xml file.
1. Modify res/values/ids.xml and place your IDs there, as given by the
   Developer Console (create the leaderboards and achievements necessary for
   the sample, if any). In the Developer console, select a resource type
   (Achievements, Events, Leaderboards) and click "Get Resources". Copy the
    contents from the console and replace the contents of res/values/ids.xml.
1. Compile and run.

IMPORTANT: make sure to sign your apk with the same certificate
as the one whose fingerprint you configured on Developer Console, otherwise
you will see errors.

IMPORTANT: if you are testing an unpublished game, make sure that the account you intend
to sign in with (the account on the test device) is listed as a tester in the
project on your Developer Console setup (check the list in the "Testing"
section), otherwise the server will act as though your project did not exist and
return errors.

<h3>If you're using another build system...</h3>

If you are using your own build system, here is a summary of what you must do:

1. Configure it to treat **google-play-services_lib** and **BaseGameUtils** as library projects, which means that not only their code but also their resources will also get added to the final build.
1. Make sure **TrivialQuest** depends on **BaseGameUtils**
1. Make sure **BaseGameUtils** depends on **google-play-services_lib**.
1. Make sure the build system is signing the APK with the right certificate (the one whose fingerprint you provided in the Developer Console when creating your client ID)

<h3>Building</h3>
To build the samples after you have applied the changes above, you can use the build/run option in
Eclipse or Android Studio, or build directly from the command line if you prefer:

```
cd /path/to/BasicSamples
./gradlew build
```